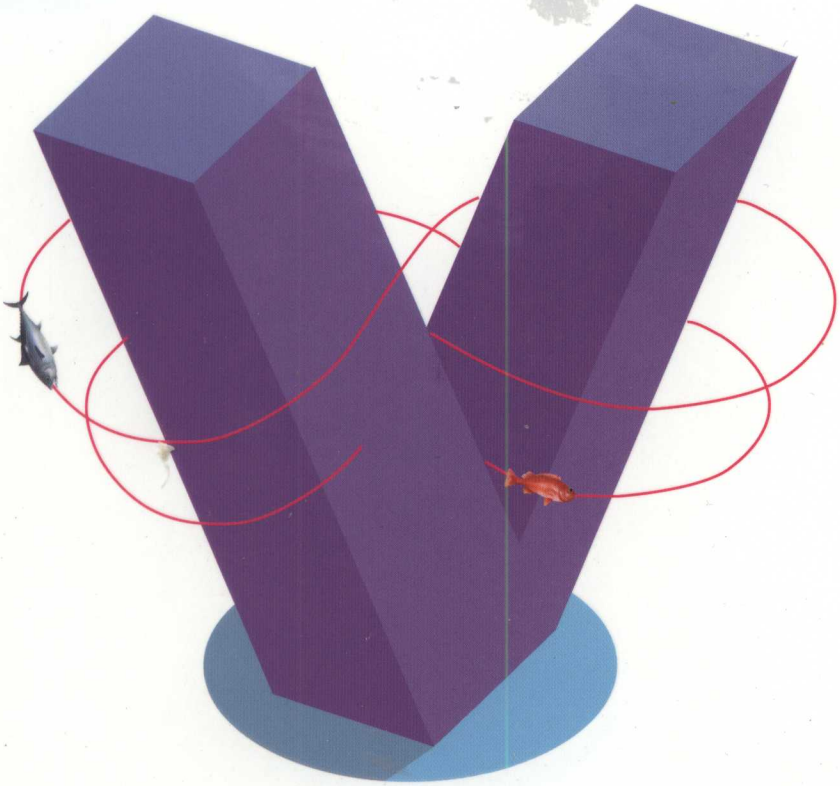
RWD

Bootstrap
Foundation
Pure 的
学习与使用

响应式网页设计实战

丰富的范例程序和详细的图解
逐一介绍响应式网页设计的
核心技术和方法

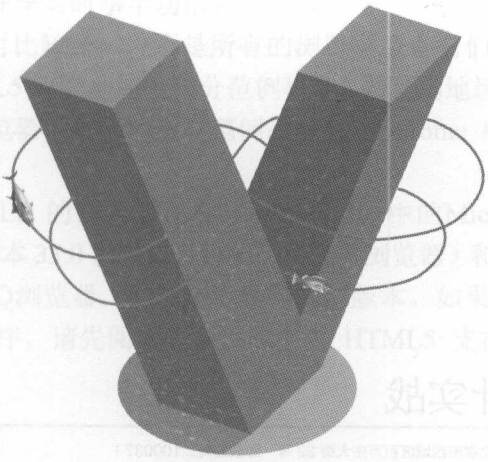
柯霖廷 等编著



机械工业出版社
China Machine Press

响应式
网页设计实战

柯霖廷 等编著



机械工业出版社
China Machine Press

图书在版编目（CIP）数据

响应式网页设计实战 / 柯霖廷等编著. —北京：机械工业出版社，2016.10

ISBN 978-7-111-54942-0

Ⅰ. ①响… Ⅱ. ①柯… Ⅲ. ①网页制作工具 Ⅳ. ①TP393.092.2

中国版本图书馆CIP数据核字（2016）第231156号

自适应/响应式网页设计（RWD）就是集中创建页面的图片排版大小，智能地根据用户行为以及使用的设备环境（系统平台、屏幕尺寸、屏幕定向等）进行相应的布局。本书以丰富的范例程序和详细的图解逐一介绍响应式网页设计的技术架构及设计要领，使用HTML与CSS设计网站的基本策略与技术，以及响应式网页设计的模块应用，具体内容包括自适应/响应式网页设计概述、网页的新旧切版方式和字体资源、版面尺寸的固定方式与弹性方式、Bootstrap 的学习与使用、Foundation 的学习与使用、Pure 的学习与使用。

本书内容由浅入深，同时辅以易于测试和操作范例的程序，便于读者详细了解程序的运行原理，是一本很好的程序设计参考书。

响应式网页设计实战

出版发行：机械工业出版社（北京市西城区百万庄大街 22 号　邮政编码：100037）

责任编辑：夏非彼　迟振春

印　　刷：中国电影出版社印刷厂　　版　　次：2016 年 10 月第 1 版第 1 次印刷

开　　本：188mm × 260mm　1/16　　印　　张：15

书　　号：ISBN 978-7-111-54942-0　　定　　价：49.00 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：（010）88379426　88361066　　投稿热线：（010）88379604

购书热线：（010）68326294　88379649　68995259　　读者信箱：hzit@hzbook.com

序 1

响应式网页设计就是本书的核心。在智能手机、平板电脑等移动设备大行其道的时候，网页设计人员如果不会灵活和熟练地运用响应式网页设计技术将电脑桌面网页设计技术拓展到移动设备上，那么不但自己会疲于奔命，而且最终会被市场淘汰。

本书以丰富的范例程序和详细的图解逐一讲解响应式网页设计的核心技术和方法，这种网页设计技术和方法会根据用户所使用设备的浏览器环境（例如屏幕的长度、宽度、长宽比、分辨率或设备屏幕显示的方向等）自动调整网页的版面，将恰当的内容和最佳的显示结果提供给用户。简而言之，就是网页设计人员只要设计统一版本的网页程序，就能在电脑、智能手机和平板电脑等各种设备上完整展示网页内容，而无需为不同屏幕大小和功能的设备分别设计和改写网页程序。

本书涉及的内容包括HTML4、HTM5、JavaScript和CSS，读者需要具备一些相关方面的知识，才能在学习时事半功倍。

HTML5 标准相对比较新，并不是所有的浏览器或者它们的旧版本都能完整地支持本书中含有HTML5 语言功能的部分范例程序。要正确地运行这些范例程序，建议使用最新版本的浏览器，本书的所有范例程序都在Chrome 版本 47（谷歌的浏览器）上调试并通过。

当然，支持HTML5 的浏览器还有Windows 10 中的Microsoft Edge、Internet Explorer 11、Opera 版本 33.0 以上以及Firefox（火狐浏览器）和Safari等的最新版本。另外，还有国内的 QQ浏览器、猎豹浏览器等的新版本。如果读者想在这些浏览器上运行 HTML5 的程序，请先阅读版本说明中对 HTML5 支持程度的细节说明。

赵　军

2016 年 8 月

序 2

本书与业界其他相关书的不同之处在于，其各章节涉及的内容都和响应式网页设计的技术架构有关。第 1 章是关于自适应 / 响应式网页设计的概述，第 2 章介绍网页新旧切版的方式和字体资源，第 3 章介绍版型尺寸的固定与弹性，第 4 章介绍 Bootstrap 的学习与使用，第 5 章介绍 Zurb's Foundation 的学习与使用，第 6 章介绍 Yahoo's Pure 的学习与使用。

为了能更好地理解第 2 章开始的各个范例程序，读者需要具备HTML、CSS 和 JavaScript 的相应知识。若没有这些基础知识，而试图直接通过各个章节的内容来理解各个范例程序，则势必会事倍功半、劳心不已……希望没有基础的初学者，先去学习上述的基础知识，有牢固的根基才好！

从第 4 章开始的内容着重于更为全面的响应式网页设计模块的应用，以指导程序设计人员更快速地创造出具有特定质感和风格的各个组件。各个组件所对应的模块原则上都被打包在外部 CSS 文件和 JS 文件中，并通过 HTML 语句在特定网页中显示出各个组件的外观。

第 4 章至第 6 章分别介绍不同厂商免费提供的组件模块，可以通过 CSS 文件和 JS 文件的链接语句，使得隶属于不同厂商的各个组件模块同时应用于同一个网页当中，再经由美编人员尽量使此网页里的各个组件模块呈现出同一质感和风格，以营造出网页中统一的整体外观。

本书章节内容的安排经由作者巧思而成，以便读者能够轻松驾驭，逐步理解本书的精髓，同时辅以易于测试和操作的范例程序，让读者详细了解程序的运行原理。相对而言，第 4 章到第 6 章的一些范例程序有些烦琐与复杂，因此以特定的程序代码区段为单位来测试其显示的组件外观，是一种较为惬意的学习方式。

本书是一本程序设计参考书，读者若觉得一大段程序代码让自己感到心烦且有压力感，不妨先看懂特定程序代码所对应的说明文字，并对自己不熟悉的英文单词做个记号，通过在线英文字典进行查询，以加快自己对程序代码的深度理解。

随着时间的推移，本书所用到的各厂商的组件模块也会改版，因此在网络云盘中提供了各章的范例程序文件，下载地址为http://pan.baidu.com/s/1hsuGdNI（注意区分字母的大小写以及数字和字母）。如果下载有问题，请电子邮件联系booksaga@126.com，邮件主题为“响应式网页设计实战”。

在此恭祝各位读者在学习上百尺竿头更进一步，不懈努力，持之以恒，蒸蒸日上，事半功倍！

编　者

目　录

第 1 章　自适应 / 响应式网页设计概述

自适应 / 响应式网页设计（RWD，responsive web design）就是指在多种设备的不同尺寸的屏幕上，精巧设计和制作出最佳视觉体验的网页。所谓的最佳视觉体验，是指：

- 用户更容易浏览网页内容。
- 用户需要更少的窗口缩放、平移与滚动的操作。

1.1　弹性的样式与版面

按RWD 方式设计出来的网站，通过可按比例调整的窗格、多媒体对象（图片、视频、动画）与 CSS3 媒体查询语句，使得各个网页的整体版面可缩放自如地显示于各种视觉环境当中。举例来说：

- 可按比例调整的窗格，要求在网页中的各元素（文字、窗体、多媒体等对象）都以相对的百分比（%）来表示，而不是以像素（px，pixel）或点数（pt，point）作为尺寸单位。
- CSS3 媒体查询语句允许特定网页根据不同设备的屏幕尺寸或其他特性而应用不同的 CSS 样式规则。

1.2　以移动设备为优先

以移动设备为优先、低调的 JavaScript 与渐进强化支持RWD的方式是RWD机制在形成具体概念之前的基本意识。较早的移动设备上的浏览器并不能解读 JavaScript 语言或 CSS3 媒体查询语句，所以在早期业界较常做的方法是：对于同样内容的网页，为了支持移动设备的用户群，需要再额外制作内容大致相同的网页。

较先进的移动设备必须渐进地支持 JavaScript 语言与 CSS3 媒体查询的新语句，才能使用渐进强化的 RWD 机制。

1.3 业界流行的 RWD Framework

现在业界开始流行所谓的免费 RWD Framework（框架），可大幅减少程序设计人员在编写程序代码所花费的时间，因为这些框架已经内置了以 CSS 和 JavaScript 程序代码为基础的表格、窗体、菜单、按钮、导航栏、文本栏等元素和多种版面类型（简称版型）。

下面给出 3 个知名 RWD Framework 的官方网址。

- Twitter's Bootstrap: http://getbootstrap.com，如图 1-1 所示。

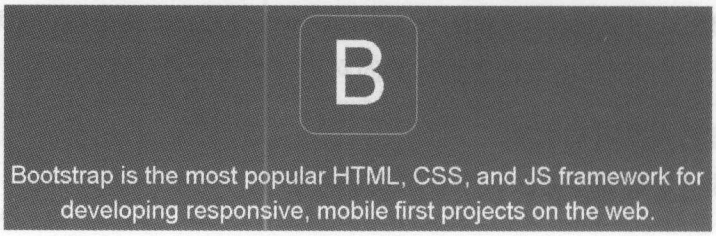


图 1-1 Bootstrap 官网上的文字介绍

- Zurb's Foundation: http://foundation.zurb.com，如图 1-2 所示。

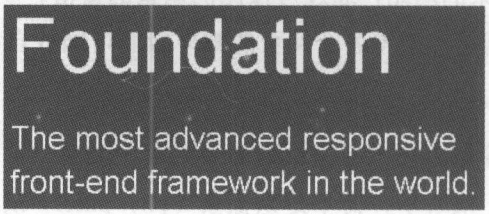


图 1-2 Foundation 官网上的文字介绍

- Yahoo's Pure: http://purecss.io，如图 1-3 所示。

图 1-3 Yahoo's Pure 官网上的文字介绍

1.4 并非一定需要 RWD Framework

程序设计人员要采用 RWD 机制，也不一定要使用各个厂商提供的 RWD Framework。只要充分熟悉HTML4、HTML5、CSS2、CSS3 与JavaScript相关语句，也可以设计出具有独特 RWD 机制的网页。设计实现具有良好 RWD 机制的网页，大致需要具备以下技术（见图 1-4）。

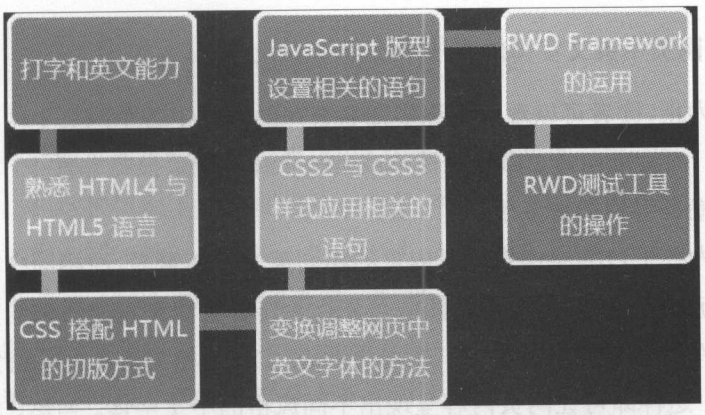


图 1-4 要设计实现 RWD 机制的网页需要具备的技术

1. 打字和英文能力

根据笔者近 10 年以来的教学经验，程序设计初学者的英文能力通常一般，可偏偏任何一种程序设计语言，无论是关键字、保留字还是内置的变量和函数等的命名，都大量采用英文！

在这种情况下，英文水平一般的初学者，在编写程序代码的过程中，会不断地拼错各种名称的英文单词，甚至在大小写字母上出错，导致其程序代码无法正常运行。

笔者也常常遇到一些初学者，一边对比讲师所提供的“相当简短”的范例程序代码，一边检查他们自己程序代码的拼字错误，却仍然检查不出问题所在。所以，有些初学者常因为自己粗心的拼写错误，再加上本身的英文能力一般，而在学习程序设计上吃尽了苦头！

2. 熟悉 HTML4 与 HTML5 语言

HTML4 和 HTML5 在语言结构上只有细微的调整，但是在元素语句上，HTML5 则增加了不少！无论是语言结构上的调整，还是元素语句上的增加，都得靠各个厂商对自家的浏览器进行改版或者升级，才能支持这些新增的部分。

Chrome、Firefox、Opera、Safari 浏览器的默认设置都会在特定的时间点上提示用户进行更新。唯独 IE 浏览器有一些麻烦的问题，对于低于 IE 9 的版本，IE默认都不会自动更新；只有IE 10 开始的版本，默认才会进行自动更新。

浏览器本身会提示用户进行更新，只有更新之后才有能力持续支持 HTML、CSS 与 JavaScript 的新语句。

3. CSS 搭配 HTML 的切版方式

早期的网页排版方式主要是通过表格元素来实现的，近年来的网页排版方式则主要通过 CSS 搭配 HTML 的切版方式来实现。这样的做法有明显的优点，可使得特定网页易于删减、扩充与调整版面，而主流浏览器的各个厂商也花费了不少心力支持这一项技术。

4. 变换调整网页中英文字体的方法

到 2015 年为止，读者可以观察到许多知名公司的网站中各个网页上中文采用的字体还是以新细明体居多，而标准楷体、微软雅黑体或其他字体的网页显得非常少。但是，各个网页英文字体上的表现就显得种类繁多了！

这主要是因为各平台的操作系统所内置的相同英文字体繁多，而相同中文字体非常少的缘故。为了弥补这方面的不足，网页设计团队可先行上传各主流浏览器所共同支持的特定中文字体文件到网页文件所存放的服务器中。接着再通过 CSS3 相关语句，让特定网页链接到相关的中文字体文件，这样即可让网页内容呈现出不同风格的中文字体。当然，浏览器在显示网页内容的速度上会因而变得稍微慢一点，这是网页设计团队必须权衡的问题。

5. CSS2 与 CSS3 样式应用相关的语句

CSS2 与 CSS3 在语言结构上大致相同，但是在规则语句上，CSS3 则增加了一些，而且在语言结构上稍微不同。各主流浏览器在前几年就开始相继支持 CSS3 中大部分的规则语句，设计团队可以安心在各网页的元素中应用 CSS3 的动画特效和版型外观相关的规则语句。

6. JavaScript 版型设置相关的语句

只要是 CSS 规则语句可表达的版型与外观，使用 JavaScript 一样可以办到。通过 JavaScript 来动态设置网页的版型与外观属于相当高级的方式，而且具有新增、删除、修改的高度弹性。

JavaScript不仅可以动态修改网页内各个元素的外观，还可以整体修改各个 CSS 规则程序代码。现如今各主流浏览器都已支持这项技术，因而设计团队可自由发挥的空间更加宽广。

7. RWD Framework 的运用

虽然设计团队为实现具有 RWD 机制的网页不一定非得使用厂商们所提供的 RWD Framework，但是使用 RWD Framework可以让设计团队节省开发**底层程序代码**所花费的大量时间。RWD Framework 本身提供了较为完整的底层程序代码，设计团队因此可以以 RWD Framework为基础快速地在其上层进行网页的设计。

8. RWD 测试工具的操作

阶段性设计出具有 RWD 机制的网页半成品后，设计团队还应该充分利用相关的 RWD 测试工具来检测在不同屏幕尺寸下这些网页半成品是否显示正常，并持续修正，直到无瑕疵为止。

其实，各主流浏览器都已经内置了较为直觉和易于上手的 RWD 测试工具，它们的功能大同小异。

1.5 RWD 实现的机制

一般而言，同样的信息内容可被规划成 4 种不同尺寸的网页：台式机、笔记本电脑、平板电脑与智能手机。在这种情况下，早期方式会产生 4 个独立的网页文件。假如将台式机和笔记本电脑屏幕显示视为同一种网页版本，则会产生 3 个网页文件。

如今的方式则可通过 CSS3 媒体查询语句，配合 JavaScript 相关语句，融合上述各个独立的网页文件成为统一的单个网页文件。

这样的网页内容在不同尺寸的设备屏幕上显示时大致会呈现如下方式。

- 网页各元素的尺寸被缩放。
- 网页各元素内的字体大小被缩放。
- 网页内的信息内容在大尺寸屏幕中全部被显示出来，而在小尺寸屏幕里则部分被隐藏起来，通过用户的触控，再把被隐藏起来的信息内容显示于画面当中。

1.6 浏览器的支持度

截至 2015 年 8 月，全球最流行的 5 种主流浏览器按照其使用率的排名（见表 1-1）为 Chrome、Firefox、Internet Explorer (IE)、Safari、Opera。

表 1-1 主流浏览器的使用率排名

统计年月	Chrome	IE	Firefox	Safari	Opera
2015 年 08 月	64.0 %	6.6 %	21.2 %	4.5 %	2.2 %
2014 年 12 月	61.6 %	8.0 %	23.6 %	3.7 %	1.6 %
2013 年 12 月	55.8 %	9.0 %	26.8 %	3.8 %	1.9 %
2012 年 12 月	46.9 %	14.7 %	31.1 %	4.2 %	2.1 %
2011 年 12 月	34.6 %	20.2 %	37.7 %	4.2 %	2.5 %
2010 年 12 月	22.4 %	27.5 %	43.5 %	3.8 %	2.2 %
2009 年 12 月	9.8 %	37.2 %	46.4 %	3.6 %	2.3 %
2008 年 12 月	3.6 %	46.0 %	44.4 %	2.7 %	2.4 %

上述 5 种浏览器，除了 IE 浏览器之外，默认都会进行自动更新，而且都支持 JavaScript 语言与 CSS3 媒体查询语句。IE 浏览器从第 9 版开始才支持 CSS3 媒体查询语句。另外，新版的 Safari 浏览器目前只运行在苹果公司所开发的操作系统上。所以，读者如果要测试 RWD 各阶段的成果，建议使用 Chrome、Firefox 或 Opera 浏览器。下面列出上述 3 种浏览器的下载网址。

- Chrome：http://www.google.com/chrome。该浏览器的显示画面如图 1-5 所示。

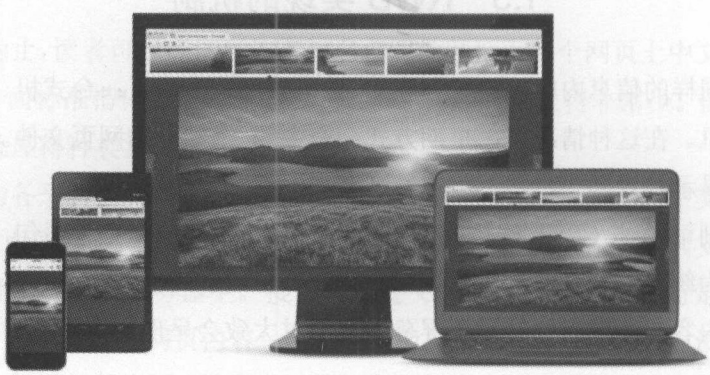


图 1-5　Chrome 浏览器在各个设备不同屏幕上显示的样子

- Firefox：http://mozilla.com.tw/firefox。该浏览器的显示画面如图 1-6 所示。

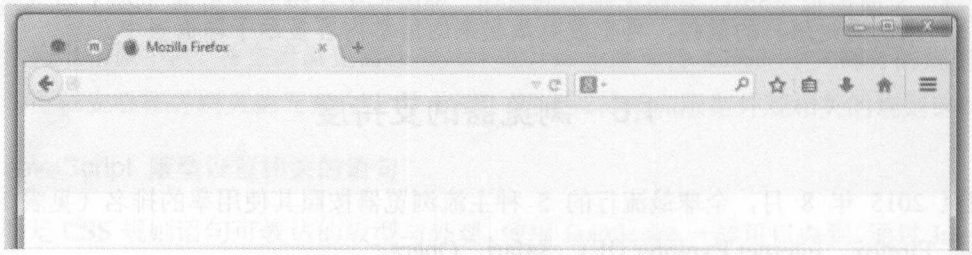


图 1-6　Firefox 浏览器的外观

- Opera：http://www.opera.com/zh-tw。该浏览器的显示画面如图 1-7 所示。

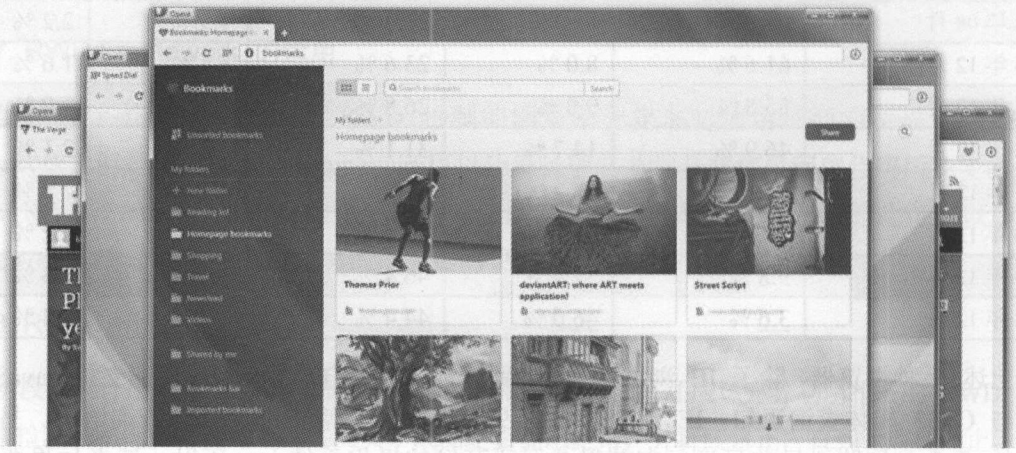


图 1-7　Opera 浏览器的外观

第 2 章　网页的新旧切版方式和字体资源

2.1　程序代码的编辑器

读者在练习本书各章节的范例时，或许已经习惯了使用某一种商业或免费的网页程序代码编辑器。而对于尚未使用这类编辑器的读者，请参考笔者推荐的两款免费编辑器，择一使用即可。

- Notepad++: http://notepad-plus-plus.org。
- Sublime Text: http://www.sublimetext.com。

其操作方式，笔者在此就不再赘述了，相信大家已经具备较为熟练的编辑器操作技能了！

2.2　CSS 搭配 HTML4 切版方式

HTML4 时代的切割版面（切版）方式被持续沿用至今，就是通过 HTML 的 div 元素语句和 CSS 相关语句来实现的切版功能。

2.2.1　开始切版前的 HTML 基本程序代码结构

为了让读者用于练习的程序代码显得更为简明扼要，笔者一开始就使用 HTML5 的基本程序代码结构作为切版的基础。

```
<!DOCTYPE html>                          让浏览器知道此为 HTML 文件。
<html>                                   html 元素的开始。
 <head>                                  head 元素的开始。
   <meta charset="UTF-8">                解读此文件内容的字符集（编码方式） 为
                                         UTF-8 万国码格式。
   <title>网页的标题</title>             设置此网页的标题。

   <style>                               style 元素的开始。
     切版程序代码关于 CSS 的部分...
   </style>                              style 元素的结束。
 </head>                                 head 元素的结束。

 <body>                                  body 元素的开始。
   切版程序代码关于 HTML 的部分...
 </body>                                 body 元素的结束。
</html>                                  html 元素的结束。
```

2.2.2 规划要创建的版面

设计团队成员要规划哪一种网页的版面，询问过项目负责人员之后，即可开始规划出目标版面。在此，笔者先以相当简单的版面为例，让读者学习如何切版。版面范例如图 2-1 所示。

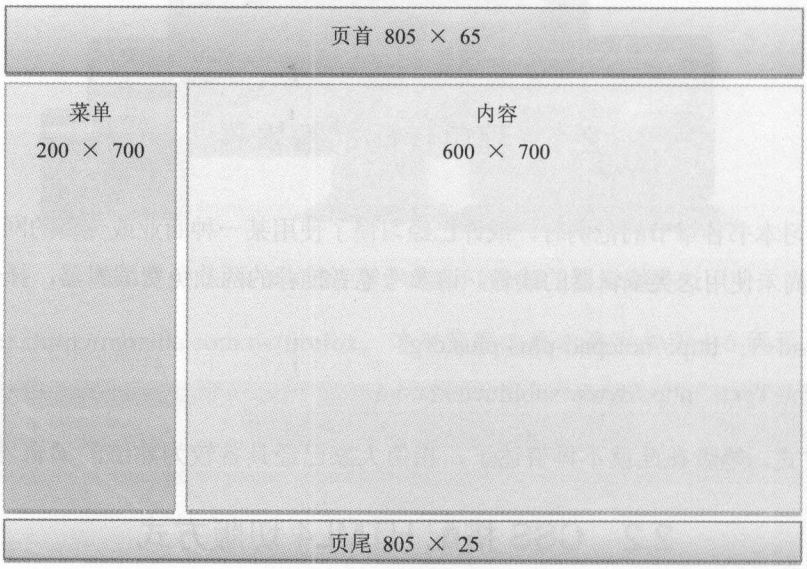


图 2-1　版面范例

在上述版面中，可以看到 4 个区块：页首、菜单、内容与页尾。每个区块间留有大致 5 个像素的间距，每个区块都被笔者标上以像素为单位的“宽度×高度”尺寸。

首先读者可在 HTML 文件中，**按照从上到下、从左到右的顺序**，提示自己要设置的区块个数与名称。

```
...
<body>
    页首
    菜单
    内容
    页尾
</body>
...
```

先把提示文字按照顺序写在 body 元素内。

在浏览器窗口中，本阶段程序代码的输出结果如图 2-2 所示。

页首 菜单 内容 页尾

图 2-2　版面输出范例 1

虽然在程序代码当中，我们是纵向编写 body 元素的内文，但是浏览器默认以横向文字来显示。

2.2.3 建立区块的程序代码

以 HTML 的 div 元素语句标上各个区块**互不相同**的身份代码（id），将原本的提示文字放置于元素语句的内文处。

```
…
<body>
   <div id="header">页首</div>
   <div id="menu">菜单</div>
   <div id="content">内容</div>
   <div id="footer">页尾</div>
</body>
…
```

每个 div 元素因此有了各自不冲突的 id 名称。请特别注意 div 元素语句具有对称性，其开始为 ，其结束为 。

从上述程序代码可以看到，每个 div 元素的 id="..." 或 id='...' 被写在其开始语句当中。对于 div 元素来说，div 是此元素的标签 / 元素名称（tag / element name），id 是此元素中的一个属性名称（attribute name），**id="..."** 中的 "..." 为 id 属性值。每个元素可以有多个属性及其对应的属性值。在浏览器窗口中，本阶段程序代码的输出结果如图 2-3 所示。

图 2-3 版面输出范例 2

div 元素是所谓的**区块**元素，其内文默认会被浏览器以**纵向排列**，这是其特性之一！

2.2.4 标示区块的范围

程序设计人员在开发阶段时，为了自我提示版面内各区块的范围，一开始最好标上不同的**背景**，以便可以清楚地看到各个区块的宽度、高度与所在的位置。

```
…
<head>
…
 <style>
   #header
   {
      background-color: DeepSkyBlue;
   }

   #menu
   {
      background-color: GoldenRod;
   }
```

#header 是 CSS 规则名称。

background-color 是 CSS 属性名称，“:”与“;”之间的是 CSS 属性值。

```
    #content
    {
       background-color: GreenYellow;
    }

    #footer
    {
       background-color: MediumPurple;
    }
  </style>
</head>

<body>
   <div id="header">页首</div>
   <div id="menu">菜单</div>
   <div id="content">内容</div>
   <div id="footer">页尾</div>
</body>
...
```

在 style 元素内，编写 CSS 规则语句，可应用特定外观样式到特定 HTML 元素上。在本例中，#header、#menu、#content 与 #footer 都是 CSS 规则名称。规则名称若以 # 开头，则表示 # 后面的是要应用此 CSS 规则的各个HTML元素，其id 属性值为header、menu、content 或 footer。

CSS 规则名称后面是一对大括号 { }，大括号里可以是一连串“CSS 属性名称: CSS 属性值;”的组合。在本例中，background-color 是属性名称（property name），而表示颜色的DeepSkyBlue、GoldenRod、GreenYellow与MediumPurple则是属性值（property value）。

HTML的 attribute 与 CSS 的 property 在翻译成中文时都为“**属性**”，请读者注意一下其语句和性质的不同。在浏览器窗口中，本阶段程序代码的输出结果如图 2-4 所示。

图 2-4　版面输出范例 3

2.2.5　设置区块尺寸

设置版面内各区块尺寸之前，应该先决定好版面的总宽度与总高度。在实际设计中来回多次调整各区块的尺寸是很常见的。

```
   ...
<style>
 #header
 {
  background-color: DeepSkyBlue;
```

```
    width: 805px;
    height: 65px;
  }

  #menu
  {
    background-color: GoldenRod;
    width: 200px;
    height: 700px;
  }

  #content
  {
    background-color: GreenYellow;
    width: 600px;
    height: 700px;
  }

  #footer
  {
    background-color: MediumPurple;
    width: 805px;
    height: 25px;
  }
</style>
  …
```

数字和表示度量单位的 px 字符之间不可以有任何空格符，而程序语句的结尾也要加上“;”，否则浏览器无法正常解读！

在特定区块元素上，通过 CSS 的 width 属性设置宽度，通过 CSS 的 height 属性设置高度。在浏览器窗口中，本阶段程序代码的输出结果如图 2-5 所示。

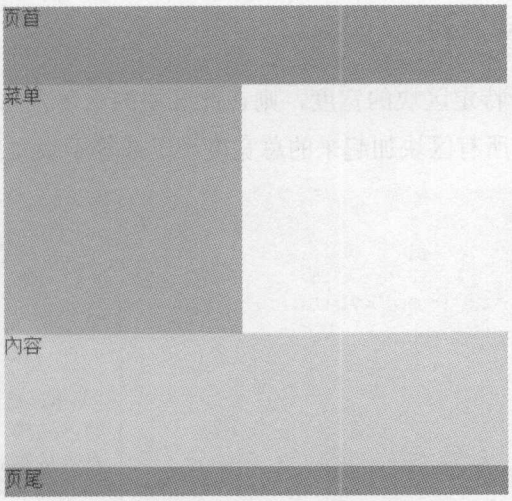


图 2-5 版面输出范例 4

可发现各区块仍旧是**纵向**排列，因而当前的版面高度为 65 + 700 + 700 + 25 = 1490 像素。不过，我们希望版面最终的高度为 65 + 5 + 700 + 5 + 25 = 800 像素！

2.2.6 限定版面总宽度

限定版面总宽度的操作是整个版面在后续调整上的关键点。后续的程序会受到此操作的影响，目前尚未有任何显示上的变化。

```
…
<style>
  #layout
  {
    width: 805px;
  }
    …
  #header
  {
    …
  }
  …
</style>
  …
<body>
  <div id="layout">
    <div id="header">页首</div>
    <div id="menu">菜单</div>
    <div id="content">内容</div>
    <div id="footer">页尾</div>
  </div>
</body>
  …
```

通过此CSS规则的应用，id为layout的 HTML元素会具有 805 像素的宽度。

通过此语句，使得 id 为 layout 的 div 元素内含原本的 4 个 div 元素，也就是 div 元素内部还有其他 div 子元素。

2.2.7 设置区块为浮动排列

若版面总宽度远大于特定区块的宽度，则在此区块的水平方向上有机会再**浮动排列**其他区块，使得水平方向上的所有区块加起来的总宽度小于或等于版面总宽度。

```
  …
#header
{
  background-color: DeepSkyBlue;
  width: 805px;
  height: 65px;
  float: left;
}

#menu
{
  background-color: GoldenRod;
  width: 200px;
  height: 700px;
```

float 属性用于设置特定 HTML 元素的浮动排列方式，其属性值为 left，表示**从左到右**浮动。

```
    float: left;
}

#content
{
  background-color: GreenYellow;
  width: 600px;
  height: 700px;
  float: left;
}

#footer
{
  background-color: MediumPurple;
  width: 805px;
  height: 25px;
  float: left;
}
  ...
```

在浏览器窗口中，本阶段程序代码的输出结果如图 2-6 所示。

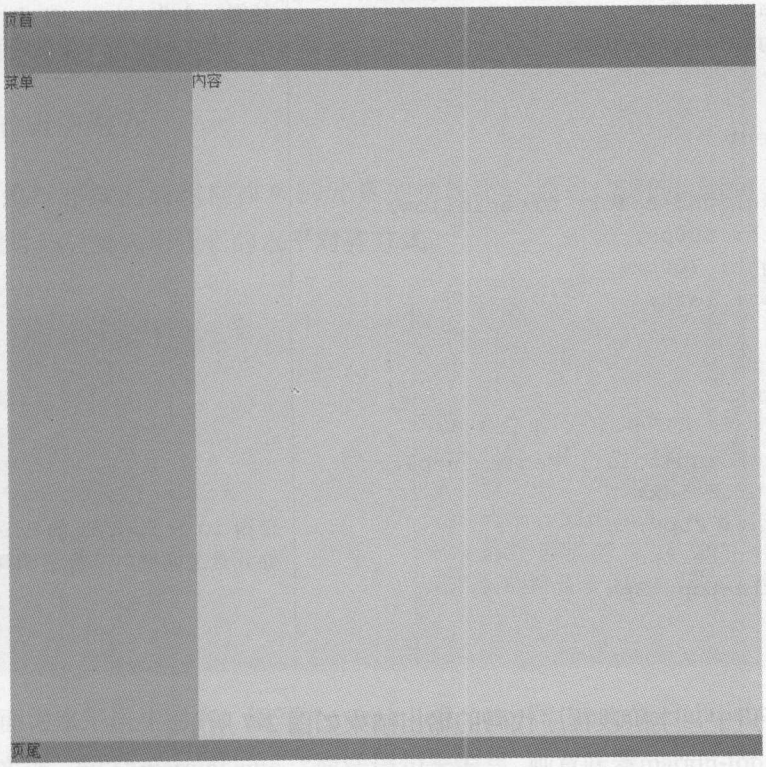


图 2-6 版面输出范例 5

经过浮动排列之后，可以看到页首与页尾区块都占满版面的总宽度，但是菜单与内容区块则**从左到右**地浮动排列在水平方向上。其实，页首与页尾区块可以不设置浮动排列。

2.2.8 调整各区块的间距

若希望在各区块之间留出较小的间距，则可通过CSS的margin、margin-top、margin-left、margin-right与margin-bottom属性来设置。

```
  …
#header
{
  background-color: DeepSkyBlue;
  width: 805px;
  height: 65px;
  float: left;
  margin-bottom: 5px;
}

#menu
{
  background-color: GoldenRod;
  width: 200px;
  height: 700px;
  float: left;
  margin-right: 5px;
}

#content
{
  background-color: GreenYellow;
  width: 600px;
  height: 700px;
  float: left;
}

#footer
{
  background-color: MediumPurple;
  width: 805px;
  height: 25px;
  float: left;
  margin-top: 5px;
}
…
```

使得 id 为 header 的 div 元素与**下方**其他元素之间空出 5 像素的间距。

使得 id 为 menu 的 div 元素与**右方**其他元素之间空出 5 像素的间距。

使得 id 为 footer 的 div 元素与**上方**其他元素之间空出 5 像素的间距。

在浏览器窗口中，本阶段程序代码的输出结果如图 2-7 所示。

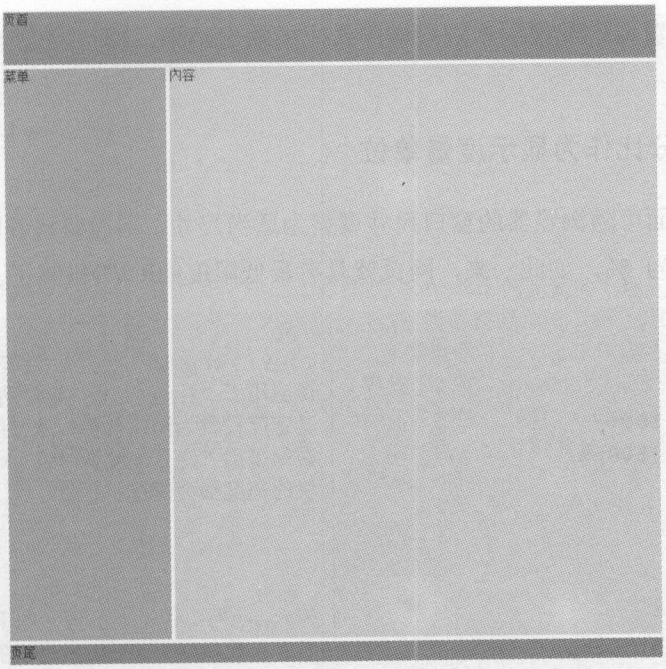


图 2-7 版面输出范例 6

可以看到各区块之间果真出现了 5 像素的间距。

2.2.9 版面的居中对齐

在没有其他一起横向或纵向排列的元素的情况下，通过CSS的margin、margin-left 或 margin-right 属性可设置应用元素的**水平对齐方式**。

```
…
#layout
{
  width: 805px;
  /*
  margin-left: auto;
  margin-right: auto;
  */
  margin: auto;
}
…
```

CSS 程序语句的注释符号，和其他许多程序语言是相同的，以“/*”符号开始，以“*/”符号结束。“/”与“*”字符之间不可以有任何空格符！CSS 的 margin 属性可用来设置特定元素和其他元素的间距。若其属性值为 auto，则表示由浏览器**自动调整**适当的间距！

要设置版面为水平居中对齐，编写“margin-left: auto;”“margin-right: auto ;”或“margin: auto ;”都可以实现。若以“margin: auto ;”简洁语句来编写，则意味着margin-top、margin-bottom、margin-left 与 margin-right 的属性值都为 auto。此外，浏览器默认不会去处理 margin-top 与 margin-bottom 的部分，因为此元素并**未被设置**明确的**高度**，所以读者看不到特定区块的上方与下方空出的相等间距。

读者在浏览器窗口中观察到本阶段程序代码的输出结果，即可发现当前的版面**水平居中**于窗口中了！

2.2.10 改用百分比作为显示度量单位

若希望网页版面可随浏览器的窗口尺寸调整为适当尺寸，则需要将各区块元素用到的 **px** 或其他度量单位改为 **%**。如此一来，网页就具有最低限度的RWD机制了。

```
...
body
{
  width: 100%;
  height: 800px;
}


#layout
{
  /* width: 805px; */
  width: 62.9%;
  height: 100%;

  /*
  margin-left: auto;
  margin-right: auto;
*/
  margin: auto;
}

#header
{
  background-color:
DeepSkyBlue;
  /* width: 805px; */
  width: 100%;
  /* height: 65px; */
  height: 8.125%;

  float: left;
  margin-bottom: 5px;
}

#menu
{
  background-color: GoldenRod;
  /* width: 200px; */
  width: 24.845%;

  /* height: 700px; */
```

body 字样前面不可加上“#”，因为此 CSS 规则应该应用于“标签/元素”名称为 body 的元素上。其宽度设置为浏览器窗口宽度的 100%，其高度仍然必须采用“绝对”度量单位 px，否则整个版面各区块的高度都会变得不正常。

因为 id 为 layout 的元素的上一层是 body 元素，所以在此是指 body 元素宽度的 62.9% 和高度的 100%。

id 为 header 的元素的上层元素是 id 为 layout 的元素，所以在此是指 id 为 layout 的元素**宽度**的 100%。

在此是指 id 为 layout 的元素**高度**的 8.125%。

id 为 menu 的元素的上层元素是 id 为 layout 的元素，所以在此是指 id 为 layout 的元素**宽度**的 24.845%！

```
        height: 87.5%;

        float: left;
        margin-right: 5px;
      }

      #content
      {
        background-color:
GreenYellow;
        /* width: 600px; */
        width: 73%;
        /* height: 700px; */
        height: 87.5%;
        float: left;
      }

      #footer
      {
        background-color:
MediumPurple;
        /* width: 805px; */
        width: 100%;

        /* height: 25px; */
        height: 3.125%;

        float: left ;
        margin-top: 5px ;
      }
      …
```

在此是指 id 为 layout 的元素**高度**的 87.5%。

id为content的元素的上层元素是id为 layout的元素，所以在此是指 id 为 layout 的元素**宽度**的73%和**高度**的 87.5%。浏览器的窗口尺寸越小时，会导致 id 为 content 的元素的实际宽度有更大的误差。原本应该设置为 600 / 805 = 74.534%，为了避免误差而导致其元素的位移，在此才被设置为 73%。

id 为 footer 的元素的上层元素是 id 为 layout 的元素，所以在此是指 id 为 layout 的元素**宽度**的 100%!

在此是指 id 为 layout 的元素**高度**的 3.125%。

以“%”作为度量单位进行改版之后，读者在浏览器窗口中任意调整窗口的宽度，并观察本阶段程序代码的输出结果，可以发现网页版面的**宽度**会随着浏览器窗口宽度的变化而调整，如图 2-8 所示。

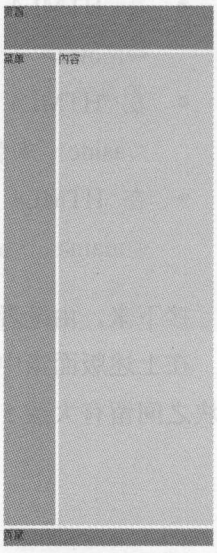


图 2-8 版面输出范例 7

2.3 CSS 搭配 HTML5 切版方式

HTML5 多了一些用来表示网页不同区块的元素语句：

- <article>…</article>：适合用作文章区块。
- <aside>…</aside>：适合用作侧边区块。
- <details>…</details>：适合用作可展开/收合的详细信息区块。
- <figcaption>…</figcaption>：适合用作图解标题区块。
- <figure>…</figure>：适合用作图解区块。
- <footer>…</footer>：适合用作页尾区块。
- <header>…</header>：适合用作页首区块。
- <main>…</main>：适合用作主要内容区块。
- <mark>…</mark>：适合用作重点区块。
- <nav>…</nav>：适合用作导航区块。
- <section>…</section>：适合用作小节区块。
- <summary>…</summary>：适合用作摘要区块。
- <time>…</time>：适合用作时间 (特定日期) 区块。

举例来说：

- 在 HTML4 中的 … 可用内置于 HTML5 的 <header> … </header> 来取代。
- 在 HTML4 中的 … 可用内置于 HTML5 的 <footer> … </footer> 来取代。
- 在 HTML4 中的 … 可用内置于 HTML5 的 <aside> … </aside> 来替代。
- 在 HTML4 中的 … 可用内置于 HTML5 的 <main> … </main>、<article> … </article> 或 <section> … </section> 来替代。

接下来，由笔者带领读者来练习制作如图 2-9 所示的网页版面。

在上述版面当中，可以看到 6 个区块：页首、横幅、菜单、内容、详细信息与页尾，每个区块之间留有大致 5 个像素的间距。

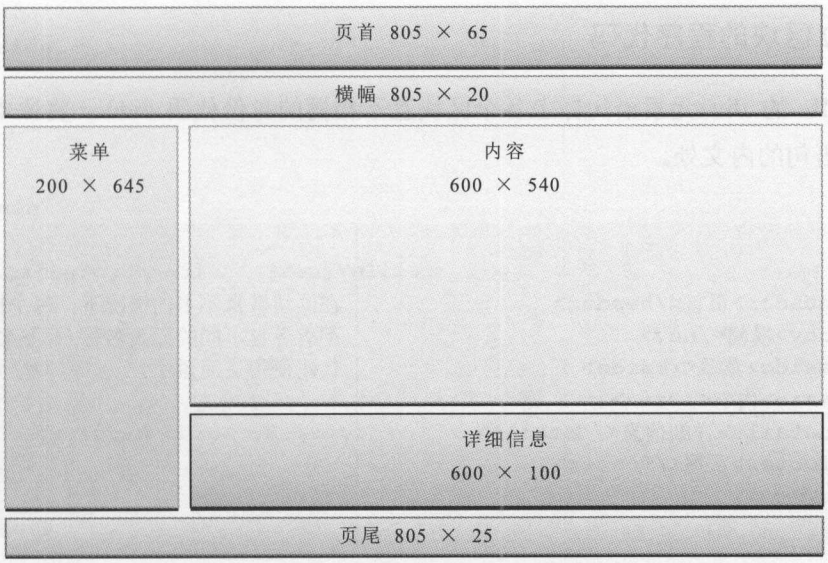


图 2-9 以此版面样式为例来制作网页

先在HTML文件中按照**从上到下、从左到右**的顺序设置区块的个数与名称：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">

    <title>HTML5 切版实例</title>

    <style>
    </style>
  </head>

  <body>
    页首
    横幅
    菜单
    内容
    详细信息
    页尾
  </body>
</html>
```

先把提示文字按照顺序写在 body 元素内。

在浏览器窗口中，本阶段程序代码的输出结果如图 2-10 所示。

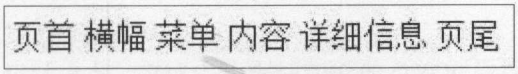


图 2-10 设置区块名称

2.3.1 建立区块的程序代码

以 HTML 的 div 元素语句标上各个区块**互不相同**的身份代码（id），将原本的提示文字放置于元素语句的内文处。

```
...
<body>
  <header>页首</header>
  <nav>横幅</nav>
  <aside>菜单</aside>
  <main>内容</main>
  <details>详细信息</details>
  <footer>页尾</footer>
</body>
...
```

在区块数量不多的情况下，每个排版用的元素都有各自不同的元素名称/标签名称，使得程序代码变得更简短了！

在浏览器窗口中，本阶段程序代码的输出结果如图 2-11 所示。

图 2-11 为各区块标上 id

从其输出结果来看，各区块除了被浏览器默认以**纵向**排列之外，details 元素所对应的区块默认为**展开**与**收合**的机制！

2.3.2 标示区块范围

从一开始就标上不同的**背景**，便可清楚地知道各个区块的宽度、高度与所在的位置。

```
  ...
<head>
  ...
  <style>
    header
    {
      background-color: DeepSkyBlue;
    }

    nav
    {
      background-color: Aquamarine;
    }
```

header 是 CSS 规则名称，其前面并没有“#”，表示此规则会应用至 HTML 的元素名称/标签名称为 header 的元素上。

```
    aside
    {
      background-color: GoldenRod;
    }

    main
    {
      background-color: GreenYellow;
    }

    details
    {
      background-color: Silver;
    }

    footer
    {
      background-color: MediumPurple ;
    }
  </style>
</head>

<body>
  <header>页首</header>
  <nav>横幅</nav>
  <aside>菜单</aside>
  <main>内容</main>
  <details>详细信息</details>
  <footer>页尾</footer>
</body>
  …
```

在浏览器窗口中，本阶段程序代码的输出结果如图 2-12 所示。

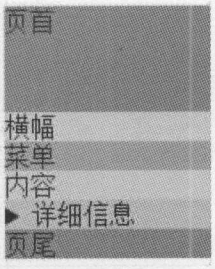


图 2-12　标示区块范围

2.3.3　设置区块尺寸

在设置版面内各区块尺寸之前，应该先决定好版面的总宽度与总高度，来回多次调整各区块尺寸是常见的情况。

```
  …
<style>
  header
  {
    background-color: DeepSkyBlue;
    width: 805px;
    height: 65px;
  }

  nav
  {
    background-color: Aquamarine;
    width: 805px;
    height: 20px;
  }

  aside
  {
    background-color: GoldenRod;
    width: 200px;
    height: 645px;
  }

  main
  {
    background-color: GreenYellow;
    width: 600px;
    height: 540px;
  }

  details
  {
    background-color: Silver;
    width: 600px;
    height: 100px;
  }

  footer
  {
    background-color: MediumPurple;
    width: 805px;
    height: 25px;
  }
</style>
  …
```

数字和表示度量单位的px字符之间不可以有任何空格符，而程序语句的结尾则要加上“;”，否则浏览器无法正常解读。

在浏览器窗口中，本阶段程序代码的输出结果如图 2-13 所示。

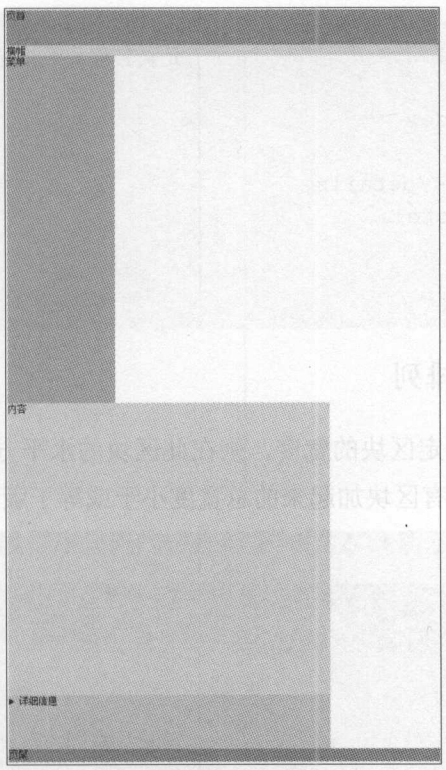


图 2-13 设置区块尺寸

各区块尚未被**浮动排列**之前，默认为**纵向排列**，因而当前的版面高度为 65 + 20 + 645 + 540 + 100 + 25 = 1395 像素。但是，我们希望版面最终的高度为 65 + 5 + 20 + 5 + 645 + 5 + 25 = 770 像素。

2.3.4 限定版面的总宽度

在本范例的后续程序里才能看出其效果，当前尚未有任何显示上的变化。

```
  …
<style>
  #layout
  {
    width: 805px ;
  }
  …
  header
  {
  …
  }
  …
</style>
  …
<body>
```

通过此 CSS 规则的应用，id 为 layout 的 HTML 元素会具有 805 像素的宽度。

```
    <div id="layout">
      <header>页首</header>
      <nav>横幅</nav>
      <aside>菜单</aside>
      <main>内容</main>
      <details>详细信息</details>
      <footer>页尾</footer>
    </div>
  </body>
  …
```

通过此语句，使得 id 为 layout 的 div 元素内含元素名称各不相同的 6 个元素。

2.3.5 设置区块为浮动排列

若版面总宽度远大于特定区块的宽度，则在此区块的水平方向上有机会再**浮动排列**其他区块，使得水平方向上的所有区块加起来的总宽度小于或等于版面总宽度。

```
  …
header
{
  background-color: DeepSkyBlue ;
  width: 805px ;
  height: 65px ;
  float: left ;
}

nav
{
  background-color: Aquamarine ;
  width: 805px ;
  height: 20px ;
  float: left ;
}

aside
{
  background-color: GoldenRod ;
  width: 200px ;
  height: 645px ;
  float: left ;
}

main
{
  background-color: GreenYellow ;
  width: 600px ;
  height: 540px ;
  float: left ;
}

details
```

表示**从左到右**浮动。

```
{
  background-color: Silver ;
  width: 600px ;
  height: 100px ;
  float: left ;
}

footer
{
  background-color: MediumPurple ;
  width: 805px ;
  height: 25px ;
  float: left ;
}
…
```

在浏览器窗口中，本阶段程序代码的输出结果如图 2-14 所示。

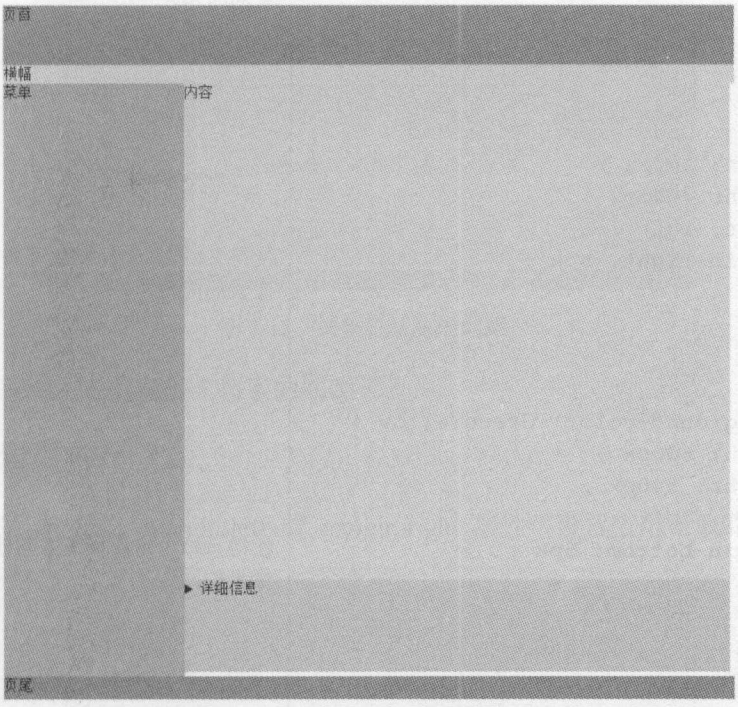


图 2-14 设置区块为浮动排列

经过浮动排列之后，可以看到页首与页尾区块都占满版面总宽度；但是菜单、内容与详细信息区块则先**从左到右**再**从上到下**地浮动排列。其实，页首与页尾区块可以不设置浮动排列。

2.3.6 调整各区块的间距

通过 CSS 的 margin、margin-top、margin-left、margin-right 与 margin-bottom 属性来设置各区块的间距。

```
…
header
{
  background-color: DeepSkyBlue ;
  width: 805px ;
  height: 65px ;
  float: left ;
  margin-bottom: 5px ;
}

nav
{
  background-color: Aquamarine ;
  width: 805px ;
  height: 20px ;
  float: left ;
  margin-bottom: 5px ;
}

aside
{
  background-color: GoldenRod ;
  width: 200px ;
  height: 645px ;
  float: left ;
  margin-right: 5px ;
}

main
{
  background-color: GreenYellow ;
  width: 600px ;
  height: 540px ;
  float: left ;
  margin-bottom: 5px ;
}

details
{
  background-color: Silver ;
  width: 600px ;
  height: 100px ;
  float: left ;
}

footer
{
  background-color: MediumPurple ;
  width: 805px ;
  height: 25px ;
```

使得 header 元素与**下方**其他元素之间空出 5 像素的间距。

使得 nav 元素和**下方**其他元素之间空出 5 像素的间距。

使得 aside 元素和**右方**其他元素之间空出 5 像素的间距。

使得 main 元素和**下方**其他元素之间空出 5 像素的间距。

```
    float: left ;
    margin-top: 5px ;
  }
  …
```

使得 footer 元素和**上方**其他元素之间空出 5 像素的间距。

在浏览器窗口中，本阶段程序代码的输出结果如图 2-15 所示。

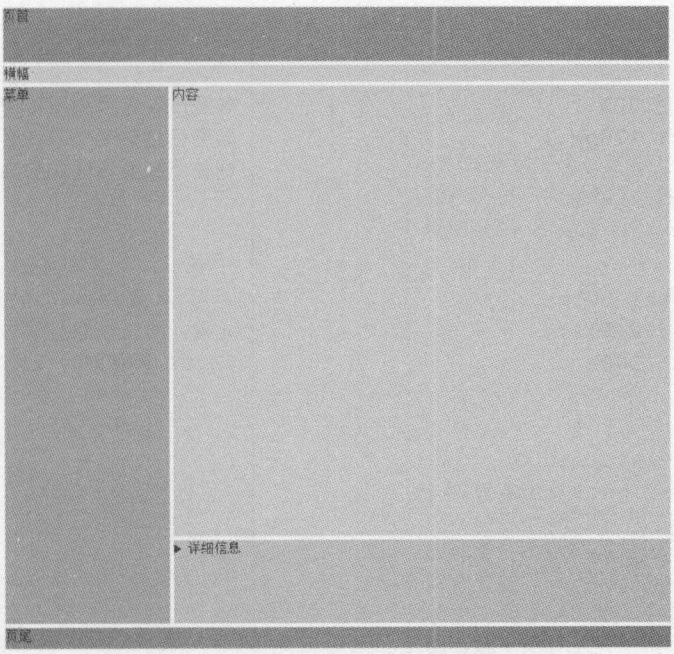


图 2-15 调整各区块的间距

可以看到各区块之间出现了 5 像素的间距。

2.3.7 版面的居中对齐

通过 CSS 的 margin 、margin-left 或 margin-right 属性可设置应用元素的**水平对齐方式**。

```
  …
  #layout
  {
    width: 805px ;

    /*
    margin-left: auto ;
    margin-right: auto ;
    */
    margin: auto ;
  }
  …
```

表示由浏览器自动调整适当的间距。

读者在浏览器窗口中观察本阶段程序代码的输出结果，可以看到当前版面**水平居中**于窗口当中了！

2.3.8 改用以百分比为显示度量单位

将各区块元素使用到的 px 或其他度量单位改成 %。如此一来，此网页就具有最低限度的 RWD 机制了。

```
…
body
{
  width: 100% ;
  height: 770px ;
}

#layout
{
  /* width: 805px ; */
  width: 62.9% ;
  height: 100% ;

  /*
  margin-left: auto ;
  margin-right: auto ;
*/
  margin: auto ;
}

header
{
  background-color: DeepSkyBlue ;
  /* width: 805px ; */
  width: 100% ;
  /* height: 65px ; */
  height: 8.442% ;

  float: left ;
  margin-bottom: 5px ;
}

nav
{
  background-color: Aquamarine ;
  /* width: 805px ; */
  width: 100% ;
  /* height: 20px ; */
  height: 2.597% ;

  float: left ;
  margin-bottom: 5px ;
}
```

其宽度设置为浏览器窗口宽度的 100%，其高度仍然必须采用“绝对”度量单位 px，否则整个版面各区块的高度都会变得不正常。

因为 id 为 layout 的元素的上一层是 body 元素，所以在此是指 body 元素**宽度**的 62.9% 和**高度**的 100%。

header 元素的上层元素是 id 为 layout 的元素，所以在此是指 id 为 layout 的元素**宽度**的 100% 和**高度**的 8.442%。

nav 元素的上层元素是 id 为 layout 的元素，所以在此是指 id 为 layout 的元素**宽度**的 100% 和**高度**的 2.597%。

```
aside
{
  background-color: GoldenRod ;
  /* width: 200px ; */
  width: 24.845% ;
  /* height: 645px ; */
  height: 83.766% ;

  float: left ;
  margin-right: 5px ;
}

main
{
  background-color: GreenYellow ;
  /* width: 600px ; */
  width: 73% ;
  /* height: 540px ; */
  height: 70.13% ;

  float: left ;
  margin-bottom: 5px ;
}

details
{
  background-color: Silver ;
  /* width: 600px ; */
  width: 73% ;
  /* height: 100px ; */
  height: 12.987% ;
  float: left ;
}

footer
{
  background-color: MediumPurple ;
  /* width: 805px ; */
  width: 100% ;
  /* height: 25px ; */
  height: 3.247% ;

  float: left ;
  margin-top: 5px ;
}
…
```

aside 元素的上层元素是 id 为 layout 的元素，所以在此是指 id 为 layout 的元素**宽度**的 24.845%和**高度**的 83.766%。

main 元素的上层元素是 id 为 layout 的元素，所以在此是指 id 为 layout 的元素**宽度**的 73% 和**高度**的 70.13%。

details 元素的上层元素是 id 为 layout 的元素，所以在此是指 id 为 layout 的元素**宽度**的 73% 和**高度**的 12.987%。浏览器的窗口尺寸越小时，会导致 details 元素的实际宽度有更大的误差！原本应该设置为 600 / 805 = 74.534%，为了避免误差而导致 details 元素的位移，在此才设置为 73%。

footer 元素的上层元素是 id 为 layout 的元素，所以在此是指 id 为 layout 的元素**宽度**的 100% 和**高度**的 3.247%。

用“%”作为度量单位进行改版之后，读者在浏览器窗口中任意调整窗口的宽度并观察本阶段程序代码的输出结果，可以发现网页版面的**宽度**会随着浏览器窗口宽度的改变而调整，如图 2-16 所示。

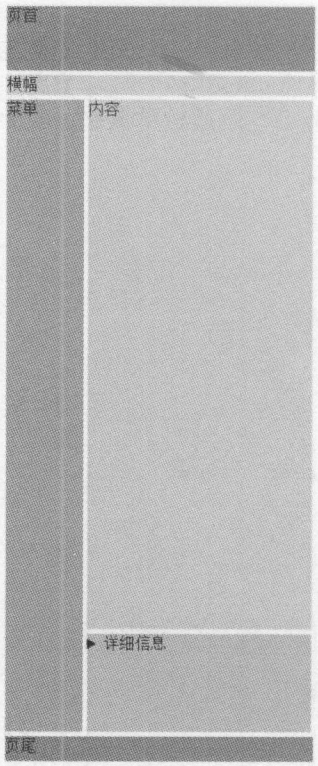


图 2-16　改用以百分比为显示度量单位

2.4　CSS 搭配 JavaScript 切版方式

经过前面两节的洗礼，读者应该已经充分了解了如何通过 HTML4、HTML5 与 CSS 相关语句来进行网页切版的工作了。笔者在此就通过 JavaScript 语句来改写前面两节所完成的网页版面。

2.4.1　精简原有的程序代码

在进行改版之前，笔者先将 2.2 节网页版面部分的 CSS 源代码中的注释删除，并将多余的空白行也删除掉，以精简和缩短程序。

```
<!DOCTYPE html>
<html>
   <head>
     <meta charset="UTF-8">

     <title>HTML4 切版实例</title>

     <style>
     body
```

```
{
  width: 100% ;
  height: 800px ;
}

#layout
{
  width: 62.9% ;
  height: 100% ;
  margin: auto ;
}

#header
{
  background-color: DeepSkyBlue ;
  width: 100% ;
  height: 8.125% ;
  float: left ;
  margin-bottom: 5px ;
}

#menu
{
  background-color: GoldenRod ;
  width: 24.845% ;
  height: 87.5% ;
  float: left ;
  margin-right: 5px ;
}

#content
{
  background-color: GreenYellow ;
  width: 73.4% ;
  height: 87.5% ;
  float: left ;
}

#footer
{
  background-color: MediumPurple ;
  width: 100% ;
  height: 3.125% ;
  float: left ;
  margin-top: 5px ;
}
</style>
</head>

<body>
```

```
    <div id="layout">
      <div id="header">页首</div>
      <div id="menu">菜单</div>
      <div id="content">内容</div>
      <div id="footer">页尾</div>
    </div>
  </body>
</html>
```

2.4.2 注释掉整段 CSS 规则语句

因为准备将原本的 CSS 程序代码改写成为 JavaScript 程序代码，在此将 style元素本身和内部的程序代码通过 HTML 的注释方式（开始符号为“<!--”，结束符号为“-->”，使得浏览器不解读注释范围内的程序代码。

```
…
<!--
<style>
…
</style>
-->
…
```

注释**开始**符号的各字符之间不可以有任何空格符。

注释**结束**符号之间也不可以有任何空格符。

2.4.3 用 JavaScript 程序代码实现特定版型

因为目前已经有现成的 HTML & CSS 切版程序代码，所以把它们快速改写成 HTML & JavaScript 切版程序代码就相当简单了。

在 script 元素中，放入 JavaScript 程序代码。将 script 元素放置于 body 元素的结束标签之前，是为了让浏览器解读重要的元素之后才开始执行 JavaScript 程序代码。

```
…
<body>
<div id="layout">
<div id="header">页首</div>
<div id="menu">菜单</div>
<div id="content">内容</div>
<div id="footer">页尾</div>
</div>

<script>
document.body.style.width = '100%' ;
document.body.style.height = '800px' ;

document.getElementById('layout').style.width = '62.9%' ;
document.getElementById('layout').style.height = '100%' ;
document.getElementById('layout').style.margin = 'auto' ;

document.getElementById('header').style.backgroundColor= 'DeepSkyBlue' ;
```

```
document.getElementById('header').style.width = '100%' ;
document.getElementById('header').style.height = '8.125%' ;

document.getElementById('header').style.float = 'left' ;
document.getElementById('header').style.marginBottom = '5px' ;

document.getElementById('menu').style.backgroundColor= 'GoldenRod' ;

document.getElementById('menu').style.width = '24.845%' ;

document.getElementById('menu').style.height = '87.5%' ;
document.getElementById('menu').style.float = 'left' ;
document.getElementById('menu').style.marginRight = '5px' ;

document.getElementById('content').style.backgroundColor= 'GreenYellow' ;

document.getElementById('content').style.width = '73.4%' ;

document.getElementById('content').style.height = '87.5%' ;

document.getElementById('content').style.float = 'left' ;

document.getElementById('footer').style.backgroundColor= 'MediumPurple' ;

document.getElementById('footer').style.width = '100%' ;

document.getElementById('footer').style.height = '3.125%' ;

document.getElementById('footer').style.float = 'left' ;

document.getElementById('footer').style.marginTop = '5px' ;
</script>
</body>
</html>
```

网页程序代码经过如上改写并存盘之后，读者在测试时，即可发现其输出结果和 2.2 节最终的网页程序代码的输出结果是相同的。

2.4.4 JavaScript 版型程序代码的第 1 阶段简化

前一小节的 JavaScript 版型程序代码可以大幅简化成如下的程序代码：

```
…
<script>
document.body.style.width = '100%' ;
document.body.style.height = '800px' ;

layout.style.width = '62.9%' ;
layout.style.height = '100%' ;
```

```
layout.style.margin = 'auto' ;

header.style.backgroundColor = 'DeepSkyBlue' ;
header.style.width = '100%' ;
header.style.height = '8.125%' ;
header.style.float = 'left' ;
header.style.marginBottom = '5px' ;

menu.style.backgroundColor = 'GoldenRod' ;
menu.style.width = '24.845%' ;
menu.style.height = '87.5%' ;
menu.style.float = 'left' ;
menu.style.marginRight = '5px' ;

content.style.backgroundColor = 'GreenYellow' ;
content.style.width = '73.4%' ;
content.style.height = '87.5%' ;
content.style.float = 'left' ;

footer.style.backgroundColor = 'MediumPurple' ;
footer.style.width = '100%' ;
footer.style.height = '3.125%' ;
footer.style.float = 'left' ;
footer.style.marginTop = '5px' ;
</script>
</body>
</html>
```

可以看出，上述程序代码中所有的“document.getElementById ('元素 id 名称')”都被修改成“元素 id 名称”了，因此其程序语句变得简短多了。

2.4.5 JavaScript 版型程序代码的第2阶段简化

前一小节的 JavaScript 版型程序代码还可进一步简化为如下的程序代码：

```
…
<script>
document.body.style.width = '100%' ;
document.body.style.height = '800px' ;

with ( layout.style )
{
width = '62.9%' ;
height = '100%' ;
margin = 'auto' ;
}

with ( header.style )
{
backgroundColor = 'DeepSkyBlue' ;
```

```
width = '100%' ;
height = '8.125%' ;
float = 'left' ;
marginBottom = '5px' ;
}

with ( menu.style )
{
backgroundColor = 'GoldenRod' ;
width = '24.845%' ;
height = '87.5%' ;
float = 'left' ;
marginRight = '5px' ;
}

with ( content.style )
{
backgroundColor = 'GreenYellow' ;
width = '73.4%' ;
height = '87.5%' ;
float = 'left' ;
}

with ( footer.style )
{
backgroundColor = 'MediumPurple' ;
width = '100%' ;
height = '3.125%' ;
float = 'left' ;
marginTop = '5px' ;
}
</script>
</body>
</html>
```

上述程序代码主要通过“with (特定元素的 style 属性的引用) { … }”来更加简化其程序语句。

2.5 跨网域运用在线字体

使用数以百计免费的在线英文字体！例如，Google Fonts（https://www.google.com/fonts）就提供了 600 多套免费的英文字体，以供网友们使用。

在网络畅通的情况下，建议直接在线使用上述的英文字体。在此以 Google Fonts 网站为例。

通过较知名的浏览器连接到 Google Fonts 页面，如图 2-17 所示。

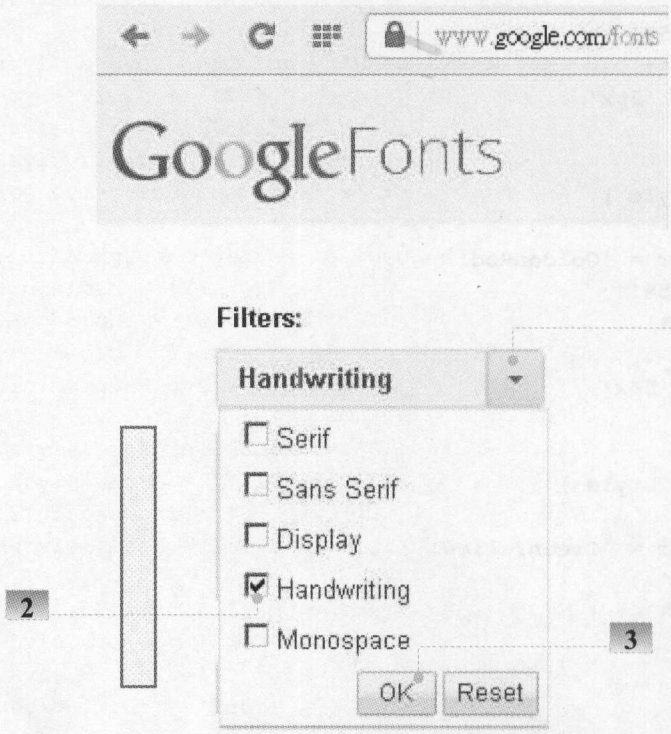


图 2-17　Google Fonts 页面和字体的选择

在页面右侧窗格当中，滚动垂直滚动条或是滚动鼠标中间按钮挑选要应用的字体。在此，笔者挑选如图 2-18 所示的“Pacifico”字体，并且单击“Quick-use”（快速应用）按钮，如图 2-19 所示。

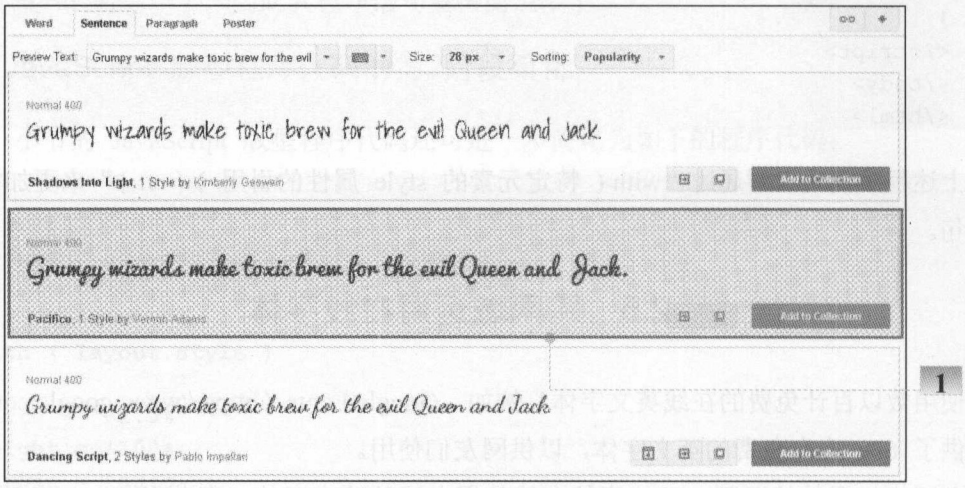


图 2-18　选择“Pacifico”字体

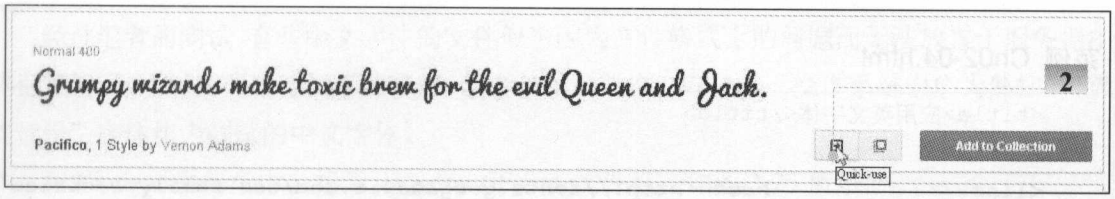


图 2-19　单击“Quick-use”（快速应用）按钮

接着按照网站上所提示的 4 个步骤操作即可！前两个步骤在大部分情况下，其选项默认都会被勾选，所以也不需要程序设计人员进行什么实际的操作。后两个步骤则需要程序设计人员将所提及的程序代码粘贴到网页程序中的特定位置上。这 4 个步骤如图 2-20 到图 2-23 所示。

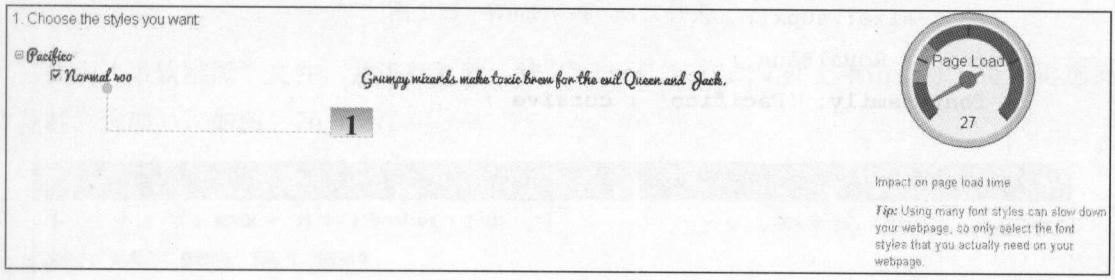


图 2-20　步骤一

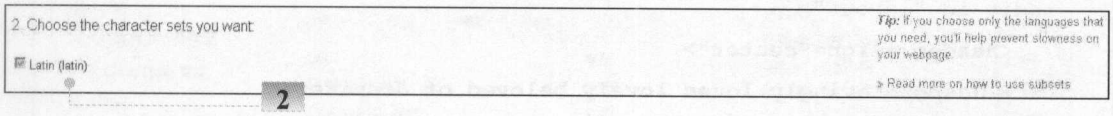


图 2-21　步骤二

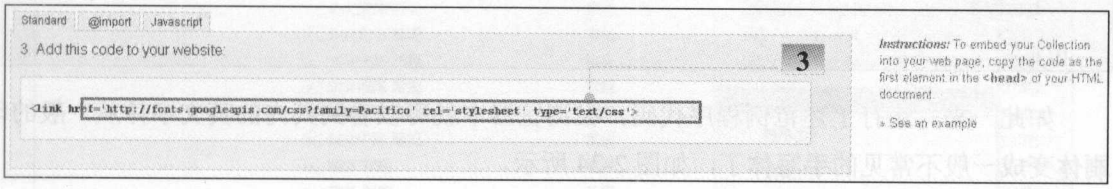


图 2-22　步骤三

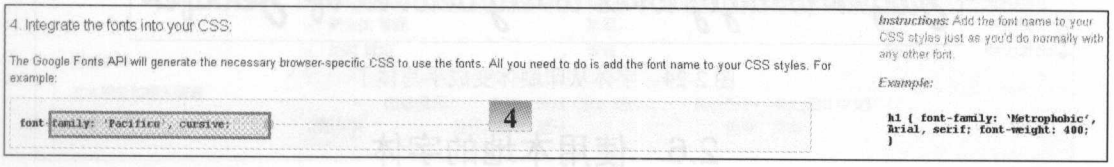


图 2-23　步骤四

在如下的范例程序代码中，被标记了底色的即是遵照上述说明步骤三和四所进行的操作。

范例 Ch02-04.html

```
    <title>应用英文字体</title>

    <link              href='http://fonts.googleapis.com/css?family=Pacifico'
       rel='stylesheet' type='text/css'>

    <style>
      header
      {
        font-size: 40px ;
        color: RoyalBlue ;
        font-family: 'Pacifico' , cursive ;
      }
    </style>
  </head>

  <body>
    <div id="layout">
      <header align="center">
        Jasper lovingly loves lovely beloved of Jennifer
      </header>
    </div>
  </body>
</html>
```

如此一来，运行上述范例程序代码，在浏览器中就可以看到其内的英文字体从一般的**印刷体**变成一般不常见的**手写体**了，如图 2-24 所示。

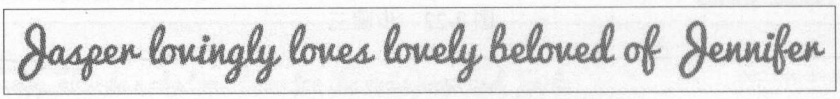


图 2-24　字体从印刷体变成手写体了

2.6　使用本地的字体

使用本地字体最常见的时机大致是程序设计人员应用“中文”字体的时候。中文字体的文件比英文字体的文件大得多，将中文字体的文件和网页文件放置于同一台服务器中，就可以**稍稍加速**特定网页的显示速度。

经过笔者的测试，有些中文字体的文件似乎因为文件格式上的问题而无法放置于服务器中被正常使用，所以需要进行如下的步骤。笔者使用的是在 Windows 操作系统中广为熟知的“微软雅黑”字体作为测试的中文字体。

到Windows系统盘中，找到 Windows/Fonts 文件夹，如图 2-25 所示。

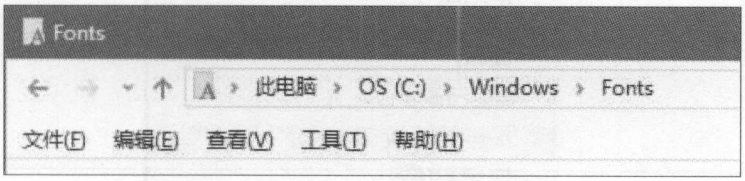


图 2-25 Windows/Fonts 文件夹

找到“微软雅黑”文件，并进行复制（按 Ctrl + C 或在此文件上单击鼠标右键，再选择“复制”选项），如图 2-26 所示。

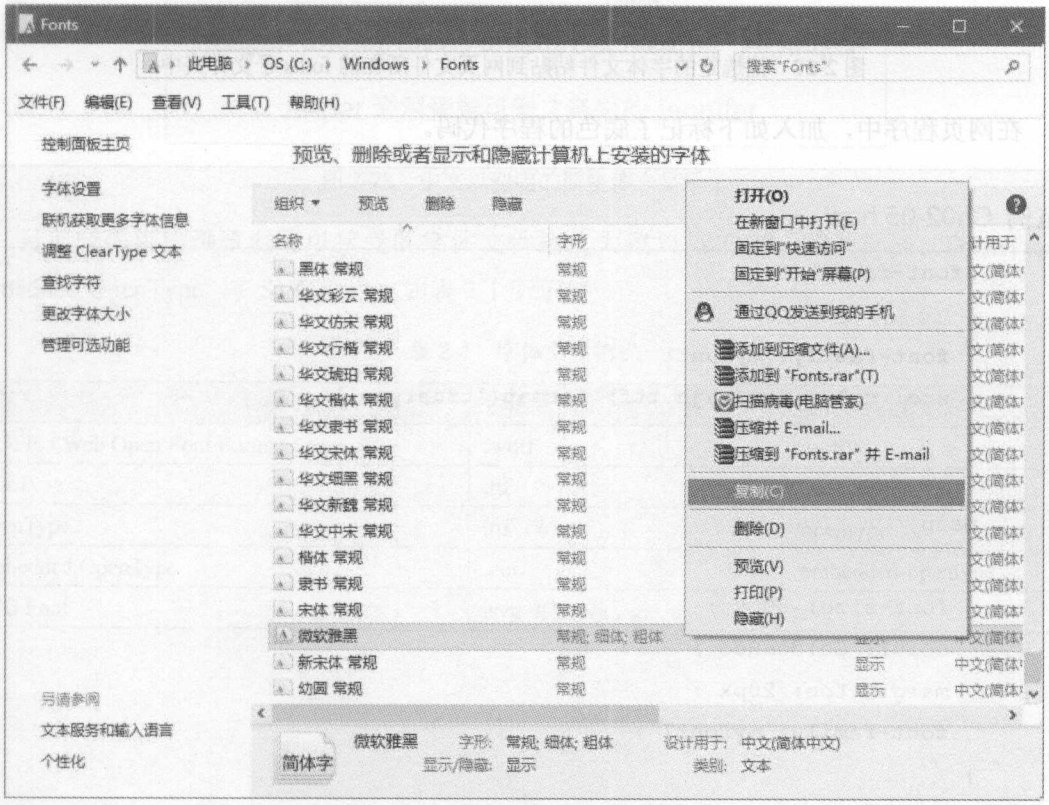


图 2-26 复制“微软雅黑”字体文件

粘贴上述中文字体的文件到网页文件所在的文件夹或子文件夹中。在本例中，将字体文件**粘贴**到网页文件所在的 fonts 子文件夹中，之后就得到如图 2-27 所示的两个字体文件。

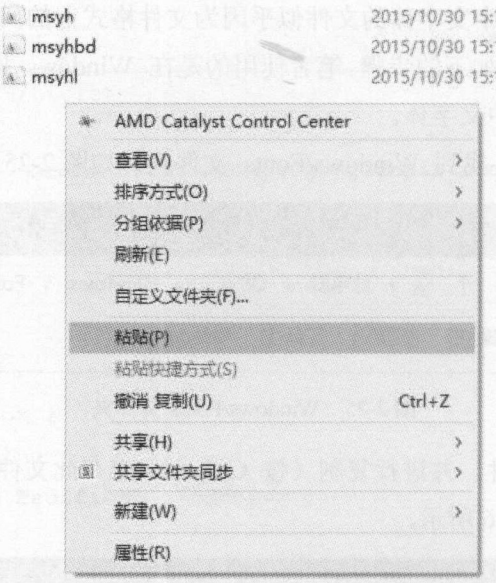


图 2-27　把选定的字体文件粘贴到网页文件所在的 fonts 子文件夹中

在网页程序中，加入如下标记了底色的程序代码。

范例 Ch02-05.html

```
    @font-face
    {
      font-family: myfont1 ;
      src: url(fonts/msjh.ttf) format('truetype') ;
    }

    main
    {
      font-size: 30px ;
      color: GoldenRod ;
      margin-top: 20px ;
      font-family: myfont1 ;
    }
  </style>
</head>

<body>
  <div id="layout">
    <header align="center">
```

```
        Jasper loving loves lovely beloved of Jennifer
      </header>

      <main align="center">
        Jasper 深深爱着可爱之亲爱的 Jennifer
      </main>
    </div>
  </body>
</html>
```

上述网页程序在浏览器的解读下，可以看到原本第二行的中文句子默认以普通的新细明体来显示，却在停顿了一下子之后，改以微软雅黑体来显示了（见图 2-28）。所以，额外指定新字体来显示网页内容时，需要给浏览器额外的反应时间。

图 2-28 中文以选定的新字体来显示了

目前较先进的浏览器可以接受的字体文件格式大致为 WOFF、TrueType、OpenType、Embedded OpenType 与 SVG Font，如表 2-1 所示。

表 2-1 字体文件格式

字体格式	扩展名	format() 指定方式
WOFF（Web Open Font Format）	.woff	'woff'
TrueType	.ttf	'truetype'
OpenType	.ttf 或 .otf	'opentype'
Embedded OpenType	.eot	'embedded-opentype'
SVG Font	.svg 或 .svgz	'svg'

第 3 章　版型尺寸的固定与弹性

3.1　HTML 传统方式

早期的 HTML 文件，因为那时的浏览器所能解读的 HTML、CSS 与 JavaScript 语句并没有太多关于页面版型尺寸支持方面的，为了使网页程序运行于不同设备的屏幕尺寸，制作的网页会被分为几个独立的网页文件，并在网页入口安排几个超链接，分别对应到这几个网页文件，由用户自行链接到最适合的网页页面。

本书的Ch03-01-desktop.html、Ch03-01-tablet.html 与 Ch03-01-phone.html 范例文件和早期的 HTML 文件不同，笔者通过 CSS 程序代码，让它们分别成为适合台式机、平板电脑和智能手机的屏幕所能显示的网页文件。

Ch03-01-phone.html 范例文件更加入了 JavaScript 程序代码，让用户在较小的屏幕尺寸中点选特定的小图标之后，才显示对应的文字内容。而在平板电脑和台式机上，这些文字内容是一开始就显示在页面当中的。

Ch03-01-desktop.html 范例文件在浏览器显示的画面如图 3-1 所示。

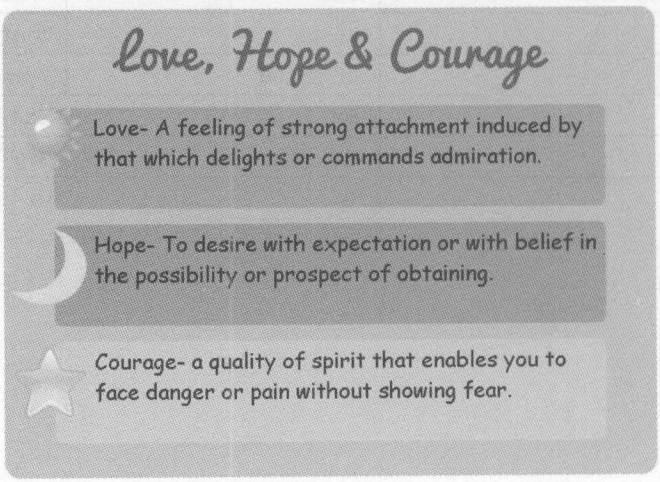


图 3-1　范例程序 Ch03-01-desktop.html 在浏览器上的显示效果

Ch03-01-tablet.html 范例文件在浏览器显示的画面如图 3-2 所示。

上述两个网页画面的不同之处在于元素范围、字体、图片的尺寸。

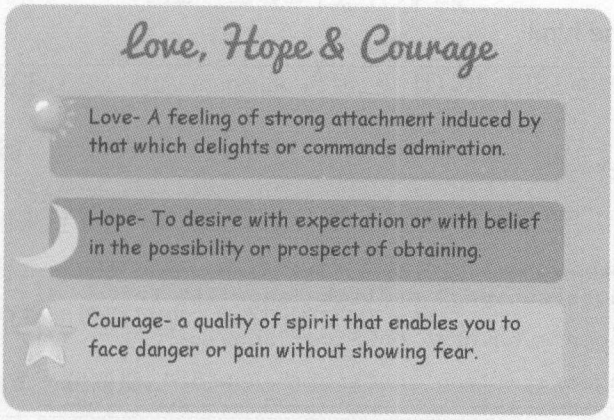


图 3-2 范例程序 Ch03-01-tablet.html 在浏览器上的显示效果

Ch03-01-phone.html 范例文件在浏览器“一开始”显示的画面如图 3-3 所示。

当用户选择太阳、月亮或星星小图片时，才会出现对应的文字内容，而被选择的前一个文字内容则会被隐藏起来，如图 3-4 所示。

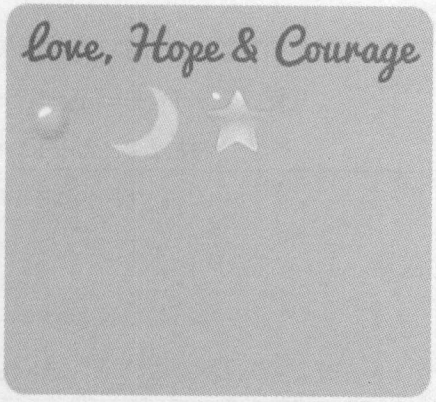


图 3-3 范例程序 Ch03-01-phone.html 在浏览器上的显示效果

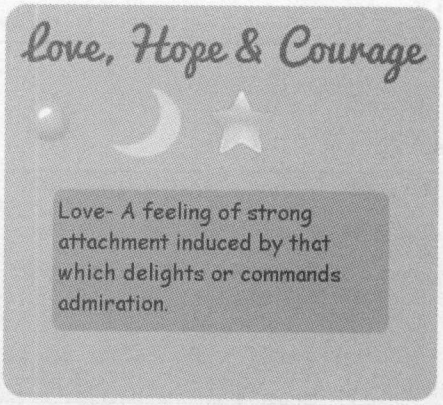


图 3-4 选择太阳、月亮或星星小图时才会出现对应的文字内容

上述网页主要受到如下标记底色的程序代码的作用。

（1）把 3 张图片的程序代码集中在一起，默认可产生小图片从左到右连续排列的效果。每个图片的语句中都带有“onClick="show(数字)"”，使得这些小图片其中之一被选择时，就会调用并执行show() 函数，并传入 1～3 的数字作为其参数。

（2）把各个图片所对应的文字内容也集中在一起，比较好决定其中哪一个会被显示出来，而其余的则被隐藏起来。

（3）此段的 JavaScript 程序代码就是用来负责**只显示 3 个文字内容中的一个，而其余两个则被隐藏起来**的机制。

范例 Ch03-01-phone.html

```
...
<body>
  <div id="layout">
    <header align="center">Love, Hope & Courage</header>

    <img src="images/sun.png" onClick="show(1)" />
    <img src="images/moon.png" onClick="show(2)" />
    <img src="images/star.png" onClick="show(3)" />

    <main class="m01">Love- A feeling of strong attachment induced by that which
        delights or commands admiration.</main>
    <main class="m02">Hope- To desire with expectation or with belief in the
       possibility or prospect of obtaining.</main>
    <main class="m03">Courage- a quality of spirit that enables you to face danger
       or pain without showing fear.</main>
  </div>

  <script>
    function show( decision )
    {
      var all_main = document.getElementsByTagName( 'main' ) ;

      for ( var i = 0 ; i < all_main.length ; i++ )
        all_main[i].style.display = 'none' ;

      document.getElementsByClassName( "m0" + decision )[0].style.display =
          'block' ;
    }
  </script>
</body>
```

1

2

3

在此稍微说明一下，“all_main[i].style.display = 'none' ;”配合 for 循环，是用来让 3 个文字内容都被隐藏起来的；而“document.getElementsByClassName ("m0" + decision) [0].style.display = 'block' ;”则是让将被选择小图片对应的文字内容显示出来的程序代码。

3.2 CSS 应用方式

本节主要讨论有关于 CSS 中的 @media 语句 和 HTML 中的 meta viewport 语句。

3.2.1 @media语句

@media 规则可用来定义在不同媒体设备上的不同样式规则。在 CSS2 时代的“媒体类型（media types）”到了 CSS3 时代，则被称为“媒体查询（media queries）”。

媒体查询可用来检查和**页面尺寸、版型相关**的如下事项。

- 浏览器窗口的宽度和高度。
- 媒体设备的宽度和高度。
- 媒体设备当前处于横向还是纵向模式。
- 屏幕的分辨率。
- 其他。

当前的主流浏览器在如下各版本之后都支持 @media 规则。

- Google Chrome 21 版。
- Internet Explorer 9 版。
- Firefox 3.5 版。
- Safari 4.0 版。
- Opera 9 版。

@media 的主要语句如下：

```
@media not | only 媒体类型 and (媒体特性)
{
    CSS 程序代码 ;
}
```

@media 另一种搭配 HTML 的 link 元素的语句可链接到不同的外部 CSS 文件：

```
<linkrel="stylesheet" media="媒体类型 and | not | only (媒体特性)"
href="外部 CSS 文件名">
```

上述**媒体类型**有以下选择。

- all: 表示所有类型的媒体设备。
- print: 表示打印机设备。
- screen: 表示计算机屏幕、平板电脑及智能型手机等设备。
- speech: 表示屏幕阅读器设备。

- aural、braille、embossed、handheld、projection、tty、tv：在 CSS3 版本里，都已被废弃，所以在这里就不再介绍了。

上述**媒体特性**有以下选择。

- aspect-ratio：指定显示区域的长宽比。
- color：指定设备在单一颜色分量的比特数（位数）。
- color-index：指定设备所能显示的颜色数量。
- device-aspect-ratio：指定设备的长宽比。
- device-height：指定设备的高度，如计算机屏幕。
- device-width：指定设备的宽度，如计算机屏幕。
- grid：指定设备是否为网格设备。
- height：指定显示区域的高度，如浏览器窗口。
- max-aspect-ratio：指定显示区域的最大长宽比。
- max-color：指定设备在单一颜色分量的最大比特数。
- max-color-index：指定设备所能显示的颜色的最大数量。
- max-device-aspect-ratio：指定设备的最大长宽比。
- max-device-height：指定设备的最大高度，如计算机屏幕。
- max-device-width：指定设备的最大宽度，如计算机屏幕。
- max-height：指定显示区域的最大高度，如浏览器窗口。
- max-monochrome：在灰度设备上，指定单一颜色最大的比特数。
- max-resolution：以 dpi 或 dpcm 为度量单位，指定设备的最大分辨率。
- max-width：指定显示区域的最大宽度，如浏览器窗口。
- min-aspect-ratio：指定显示区域的最小长宽比。
- min-color：指定设备在单一颜色分量的最小比特数。
- min-color-index：指定设备所能显示的颜色的最小数量。
- min-device-aspect-ratio：指定设备的最小长宽比。
- min-device-width：指定设备的最小宽度，如计算机屏幕。
- min-device-height：指定设备的最小高度，如计算机屏幕。
- min-height：指定显示区域的最小高度，如浏览器窗口。
- min-monochrome：在灰度设备上，指定单一颜色的最小比特数。
- min-resolution：以 dpi 或 dpcm 为度量单位，指定设备的最小分辨率。
- min-width：指定显示区域的最小宽度，如浏览器窗口。
- monochrome：在灰度设备上，指定单一颜色的比特数。
- orientation：指定设备当前是处于横向还是纵向模式。

- resolution: 以 dpi 或 dpcm 为度量单位，指定设备的分辨率。
- scan: 指定电视机是以逐行还是隔行扫描来显示图像。
- width: 指定显示区域的宽度，如浏览器窗口。

举例来说，在以下的网页文件里内含 @media 语句，可告知浏览器：当浏览器窗口的宽度超过 500 像素时，即以宝蓝色（RoyalBlue）作为背景颜色；当宽度未超过 500 像素时，则以粉红色（Pink）作为背景颜色：

范例 Ch03-02.html

```
            ...
    <style>
      body
      {
        background-color: RoyalBlue ;
      }

      @media screen and (max-width: 500px)
      {
        body
        {
          background-color: Pink ;
        }
      }
    </style>
  </head>
  <body>
    <p>浏览器窗口宽度小于 500 像素时，网页背景颜色会从宝蓝色变成粉红色！</p>
  </body>
</html>
```

再来看另一个实例，在台式机的标准屏幕尺寸之下，本范例的网页所显示的版型如图 3-5 所示，读者可以发现菜单、内容、详细信息区块都为横向连续排列的。

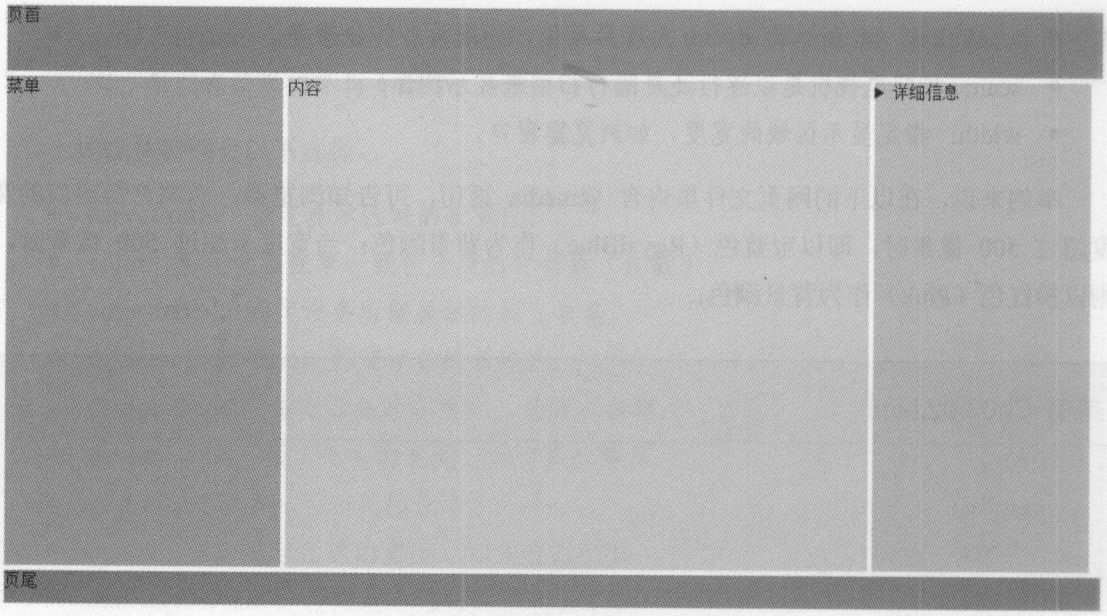


图 3-5　在台式机的标准屏幕尺寸所显示的版型

但是，一旦浏览器窗口的宽度被调整成为 660 像素以下，或是此网页显示在浏览器窗口宽度不足 660 像素的智能手机、平板电脑上时，此网页所显示的版型就会产生如图 3-6 所示的变化。

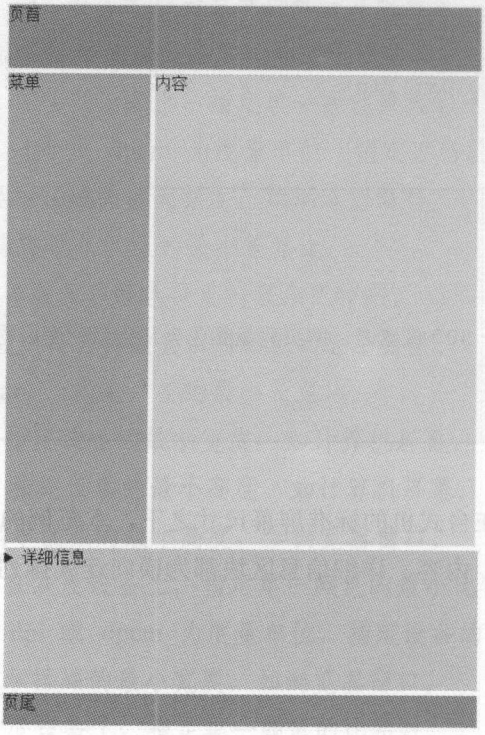


图 3-6　在浏览器窗口宽度不足 660 像素时所显示的版型

我们可以发现其版型变成**菜单、内容区块仍为横向连续排列**，但是详细信息区块出现在下方而横跨版型的总宽度。其 CSS 程序代码的安排如下：

范例 Ch03-03.html

```
    ...
<style>
  body
  {
    width: 100% ;
    height: 770px ;
  }

  #layout
   {
     width: 80% ;
     height: 100% ;
     margin: auto ;
   }

  header
  {
    background-color: DeepSkyBlue ;
    width: 100% ;
    height: 8% ;
    float: left ;
    margin-bottom: 5px ;
  }

  aside
  {
    background-color: GoldenRod ;
    width: 25% ;
    height: 60% ;
    float: left ;
    margin-right: 5px ;
  }

  main
```

```
{
  background-color: GreenYellow ;
  width: 53% ;
  height: 60% ;
  float: left ;
  margin-bottom: 5px ;
}

details
{
  background-color: Silver ;
  width: 20% ;
  height: 60% ;
  float: left ;
  margin-left: 5px ;
}

footer
{
  background-color: MediumPurple ;
  width: 100% ;
  height: 4% ;
  float: left ;
}

@media only screen and (max-width: 660px)
{
  aside { width: 30% ; }

  main { width: 67.5% ; }

  details
  {
    width: 100% ;
    height: 18% ;
    margin-left: 0px ;
    margin-bottom: 5px ;
  }
}
```

```
    </style>
  </head>

  <body>
    <div id="layout">
      <header>页首</header>
      <aside>菜单</aside>
      <main>内容</main>
      <details>详细信息</details>
      <footer>页尾</footer>
    </div>
  </body>
</html>
```

在上述程序范例中，方框内的 CSS 程序代码让浏览器得知，每当浏览器窗口宽度小于 660 像素时：

- 菜单（aside 元素）区块变成版型宽度的 30%。
- 内容（main 元素）区块变成版型宽度的 67.5%。
- 详细信息（details 元素）区块变成版型宽度的 100%，其高度变成版型高度的 18%。此外，其左侧外部间距从原本的 5 像素改成没有间距（0 像素），其下方的外部间距则从没有间距改成 5 像素的间距。

3.2.2 meta viewport语句

举例来说，“<meta name="viewport" content="width=device-width, initial-scale=1">”这条 HTML 语句的意义，读者可通过如下说明来充分了解。

上述语句的“width=”部分可用来控制屏幕视窗（viewport）的尺寸。当上述语句被设置为“width=600”时则表示视窗尺寸为 600 像素；若仍然维持“width=device-width”时，则表示视窗尺寸是**在未有任何缩放情况下以像素（pixel）为度量单位**的浏览器窗口宽度。上述语句的“initial-scale=”部分可用来控制视窗在网页一开始被加载时的缩放比例。

有些移动设备因为其屏幕显示比例的关系，有时会以 1.5 或 2 个硬件像素来显示网页中的 1 个像素，造成实质上的显示问题。“initial-scale=1”可以让硬件像素和网页显示像素尽量以 1:1 的比例来处理。

举例来说，假使特定网页的版型需要至少 500 像素的宽度，则可使用“<meta name="viewport" content="width=500, initial-scale=1">”的 HTML 语句。一旦设备屏幕宽度超过 500 像素，浏览器就不会动态改变其缩放比例，而是自动扩展视窗的宽度，以符合设备屏幕的宽度。

移动设备上的浏览器在解决当前屏幕**从纵向模式变成横向模式**的页面显示问题时，有细微的不同。例如，移动设备上的 Safari 浏览器在处理上述问题时，只会以直接比例的缩放方式来解决这样的页面显示问题。

假如程序设计人员想在这样的浏览器窗口中保持页面显示的最佳质量，则必须额外加上“, maximum-scale=1 ”的语句部分，成为“<meta name="viewport" content="initial-scale=1, maximum-scale=1">”。不过，这样也会让用户无法自行缩放所看到的网页页面。

有些移动设备上的浏览器在处理网页显示时则比较细腻，当设备的横向模式或纵向模式有所改变时，就会根据窗口的新信息来主动更新视窗尺寸、页面版型和 CSS 相关的属性值。

3.3 JavaScript 搭建方式

范例 Ch03-02.html 与 Ch03-03.html 中都只牵涉到 HTML 和 CSS 程序代码。不过，@media 的 CSS 程序代码却可以用 JavaScript 来改写。原本的 Ch03-02.html 可被改写成 Ch03-04.html，而 Ch03-03.html 则可被改写为 Ch03-05.html。

范例 Ch03-04.html

```
    ...
  <style>
    /*
    body
    {
      background-color: RoyalBlue ;
    }

    @media screen and (max-width: 500px)
    {
      body
      {
        background-color: Pink ;
      }
    }
    */
  </style>
</head>

<body onResize="change_background()">
```

```
    <p>浏览器窗口宽度小于 500 像素时，网页背景颜色会从宝蓝色变成粉红色！</p>
    <script>
      change_background() ;

      function change_background()
      {
      if ( window.innerWidth <= 500 ) document.body.style.backgroundColor =
         'Pink' ;
      elsedocument.body.style.backgroundColor = 'RoyalBlue' ;
      }
    </script>
  </body>
</html>
```

上述范例文件中，其方框中被注释掉的CSS程序代码可改用另一个方框中的JavaScript程序代码来取而代之。但是，必须在body元素的HTML语句之内加上“onResize="change_background()"”程序代码。这样才能在用户每次调整浏览器窗口尺寸时调用并执行 change_background() 函数，以通过 window.innerWidth 来获取浏览器窗口当前的宽度，以决定是否变换网页的背景颜色。

范例 Ch03-05.html

```
      ...
    <style>
      ...
      aside
      {
        background-color: GoldenRod ;
        /* width: 25% ; */
        height: 60% ;
        float: left ;
        margin-right: 5px ;
      }

      main
      {
        background-color: GreenYellow ;
        /* width: 53% ; */
        height: 60% ;
```

```
      float: left ;
      margin-bottom: 5px ;
    }

    details
    {
      background-color: Silver ;
      /*
      width: 20% ;
      height: 60% ;
      margin-left: 5px ;
      */
      float: left ;
    }
      ...
    /*
    @media only screen and (max-width: 660px)
    {
      aside { width: 30% ; }

      main { width: 68.5% ; }

      details
      {
        width: 100% ;
        height: 18% ;

        margin-left: 0px ;
        margin-bottom: 5px ;
      }
    }
    */
  </style>
</head>

<body onResize="change_layout()">
  <div id="layout">
    <header>页首</header>
```

```
    <aside id="menu">菜单</aside>
    <main id="content">内容</main>
    <details id="information">详细信息</details>
    <footer>页尾</footer>
  </div>
  <script>
    change_layout() ;

    function change_layout()
    {
      if ( window.innerWidth <= 660 )
      {
        menu.style.width = '30%' ;
        content.style.width = '67.5%' ;
        information.style.width = '100%' ;
        information.style.height = '18%' ;
        information.style.marginLeft = '0px' ;
        information.style.marginBottom = '5px' ;
      }
      else
      {
        menu.style.width = '25%' ;
        content.style.width = '53%' ;
        information.style.width = '20%' ;
        information.style.height = '60%' ;
        information.style.marginLeft = '5px' ;
        information.style.marginBottom = '0px' ;
      }
    }
  </script>
 </body>
</html>
```

在上述范例文件中，其方框内被注释掉的CSS程序代码分散于各处，可改用另一较大的方框内的JavaScript程序代码来取而代之。但是，也必须在body元素的HTML语句之内加上“onResize="change_background()"”程序代码。这样才能在用户每次调整浏览器窗口尺寸时调用change_layout() 函数并通过 window.innerWidth 来获取浏览器窗口当前的宽度，以决定是否大幅度调整网页的版型。

第 4 章　Bootstrap 的学习与使用

Twitter（推特）公司声称 Bootstrap 是用来开发自适应、以移动设备为优先的网页项目的组件，是网络上最受欢迎的HTML、CSS和JavaScript Framework。

Bootstrap 本身建立在 CSS 语言之上，而其组件则是构建在 HTML语言之上。因此，在应用Bootstrap时，是通过 HTML 元素的分层式语句，再搭配 class 属性及其属性值对应的语句来实现的。

4.1　Twitter 的 Bootstrap 简易应用方式

- 直接在线使用。新网页内的程序代码在 body 元素开始之前，加上如下语句即可：

```
    <link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/
                                                    bootstrap.min.css">

    <link rel="stylesheet"
    href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/css/
          .                                          bootstrap-theme.min.css">

    <script
    src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.5/js/bootstrap.min.js">
</script>
```

- 下载后本地使用。打开 http://getbootstrap.com/getting-started/，单击“Download Bootstrap”按钮，即可立即下载文件名类似 bootstrap-3.3.5-dist.zip 的压缩文件。解压缩该文件到存放新网页的文件夹中，解压缩后会得到名称类似 bootstrap-3.3.5-dist 的文件夹，其中包含 css、fonts 和 js 这 3 个子文件夹。各个子文件夹则包含多个 css 程序文件、eot / svg / ttf / woff / woff2 等字体文件，以及 js 程序文件。其解压缩的文件和文件夹如下：

```
bootstrap/
├── css/
│   ├── bootstrap.css
│   ├── bootstrap.css.map
│   ├── bootstrap.min.css
│   ├── bootstrap-theme.css
│   ├── bootstrap-theme.css.map
│   └── bootstrap-theme.min.css
├── js/
│   ├── bootstrap.js
│   └── bootstrap.min.js
└── fonts/
    ├── glyphicons-halflings-regular.eot
    ├── glyphicons-halflings-regular.svg
    ├── glyphicons-halflings-regular.ttf
    ├── glyphicons-halflings-regular.woff
    └── glyphicons-halflings-regular.woff2
```

其基本样板程序代码如下（Ch04-01-basic-template.html 范例文件）：

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <!-- 上述 3 个 meta 元素语句，必须在其他 meta 元素语句之前！ -->
    <title>Bootstrap 基本样板</title>

    <!-- Bootstrap 的 css 文件 -->
    <link href="bootstrap/css/bootstrap.min.css" rel="stylesheet">

    <!-- 为了使得比 IE 9 更旧的版本 IE 8 可以使用 Bootstrap，需要以下 4 行语句 -->
    <!--[if lt IE 9]>
    <script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js">
</script>
    <script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js">
</script>
    <![endif]-->
  </head>
```

```
  <body>
    <h1>World Peace!</h1>

    <!-- 对于使用 Bootstrap 的 JavaScript 插件，则需要 jQuery 函数库！ -->
    <script src="https://ajax.googleapis.com/ajax/libs/jquery/1.11.3/
                jquery.min.js"></script>
    <!-- Bootstrap 的 JS 程序文件，内含所有已被编译过的插件和需要的文件！ -->
    <script src="bootstrap/js/bootstrap.min.js"></script>
  </body>

</html>
```

请各位读者特别注意上述程序代码中的注释文字，以便了解Bootstrap基本样板中所需的固定程序代码。此外，读者也必须注意在程序代码中所提及的各个文件的路径。解压缩后，类似 bootstrap-3.3.5-dist 名称的文件夹被改名为了 bootstrap，其内的 3 个子文件夹和文件的名称并未变动。

4.2 免费小图标

Bootstrap内置 250 多个小图标，可供用户免费使用。读者可以在http://getbootstrap.com/components/ 看到它们的模样以及CSS 类名称。为了在程序中应用它们并显示于页面中，需要引用它们的 CSS 类名称，并将其写入如下的 span 元素语句当中：

```
<span class="glyphicon glyphicon-heart-empty" aria-hidden="true"></span>
```

上述语句会显示出一个没有填色的心形图标：

本节关于显示小图标和按钮的范例程序，请参考 Ch04-02-icons.html。

若要将上述小图标显示于按钮之中，可使用如下的 HTML 语句，让button按钮元素语句内含span元素语句：

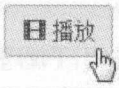
```
<button type="button" class="btn btn-default" aria-label="Left Align">
   <span class="glyphicon glyphicon-heart-empty" aria-hidden="true"></span>
</button>
```

上述语句会显示无填色的心形图标按钮，可供用户使用：

在上述按钮上会出现一个手掌形状，是因为在测试时已移入鼠标指针的关系。若要在按钮之中，除了有小图标之外，还有简短的文字，则将文字安排于span元素语句的前面或后面：

```
<button type="button" class="btn btn-default" aria-label="Left Align">
    <span class="glyphicon glyphicon-film" aria-hidden="true"></span>
    播放
</button>
```

上述语句会显示“**播放**”字样在按钮范围内“底片”小图标的右侧：



将上述的内置小图标显示于按钮之内，主要是用到了HTML的button**按钮**元素和span**行内**元素。

- button 元素是由 <button> … </button> 程序代码所构成的，用于在按钮范围内显示文字或小图标的程序代码，必须编写在 button 元素的开始语句 <button> 和结束语句 </button> 之间才行。
- 用来显示小图标的 span 元素必须编写在 button 元素之内。span 元素是所谓的行内元素，表示浏览器在网页中显示其外观之后，并不会自动换行。事实上，button 元素也是行内元素的一种。

也可以在一个信息框中的文字之前显示一个小图标，例如：

上述信息框的语句，可如下编写：

```
<div class="alert alert-danger" role="alert" style="margin: 5px ; width: 220px">
    <span class="glyphicon glyphicon-exclamation-sign" aria-hidden="true">
    </span>
    请正确输入所需的数据！
</div>
```

上述程序代码会显示出信息框，是受到“class="alert alert-danger" role="alert" ”语句的影响而应用了 Bootstrap 的信息框样式。

4.3 下拉式或上拉式菜单

4.3.1 下拉式菜单

本小节所对应的范例文件为 Ch04-03-drop-menus.html，可生成如图 4-1 所示的下拉式菜单。

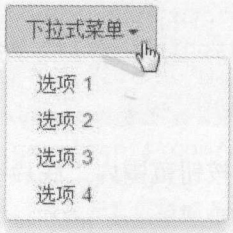


图 4-1　下拉式菜单

从图 4-1 可以看到，下拉式菜单是由一个按钮和一个开始被隐藏起来的菜单所构成的。

可以通过如下语句来制作完成这个下拉式菜单：

```
<div class="dropdown" style="margin: 10px">
 <button class="btn btn-default dropdown-toggle" type="button"
    id="dropdownMenu1" data-toggle="dropdown" aria-haspopup="true"
    aria-expanded="true">
    下拉式菜单
  <span class="caret"></span>
  </button>

  <ul class="dropdown-menu" aria-labelledby="dropdownMenu1">
    <li><a href="#">选项 1</a></li>
    <li><a href="#">选项 2</a></li>
    <li><a href="#">选项 3</a></li>
    <li><a href="#">选项 4</a></li>
  </ul>
</div>
```

在上述程序代码当中，用来启动菜单的按钮是以 type 属性值为 button 的 button 元素所构成的。为了让此 button 元素具有 Bootstrap 的**默认**外观样式，必须再额外设置其 class 属性值为 btn btn-default。

为了让此按钮元素可用来启动其对应的菜单，必须再进行以下设置。

- 替其 class 属性值加注 dropdown-toggle 字样。
- 使 data-toggle 属性值为 dropdown。
- 使 aria-haspopup 属性值为 true。
- 使 aria-expanded 属性值为 true。

为了让上述按钮中出现【倒三角形】的小图案，必须再加上其 class 属性值为 caret 的 span 元素。

被上述按钮所启动的菜单是由 ul 元素和内部各 li 元素所构成的。其每个 li 元素又包含超链接 a 元素。其中，ul 是表示无顺序编号的列表（unordered list），而 li 则是表示列表项（list item）。

本小节所提及的程序代码被 HTML 的 div 元素用来封装内部的 button 元素和 ul 元

素，而此 div 元素也通过了 class="dropdown" 语句来使用Bootstrap 的下拉式菜单样式。

div 元素是**区块**元素的一种，由浏览器显示出其外观之后会继续**换行**，然后在窗口中继续显示出其他元素的外观。

4.3.2 上拉式菜单

若要将菜单改成“上拉式”，只需将上述程序代码修改如下，即把 **dropdown** 改为**dropup**：

```
<div class="dropdown" style="margin: 10px">
↓
<div class="dropup" style="margin: 10px">
```

如此一来，菜单即可变成如图 4-2 所示的上拉式。

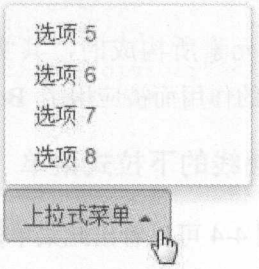


图 4-2 上拉式菜单

所谓的上拉式菜单，就是当按钮被单击之后，其对应的菜单显示在此按钮的**上方**！

4.3.3 具有选项分隔线的下拉式菜单

从图 4-3 可以看出，菜单最后两个选项之间多了一条分隔线。

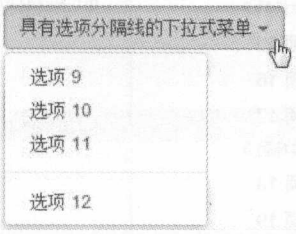


图 4-3 具有选项分隔线的下拉式菜单

```
<div class="btn-group">
    <button type="button" class="btn btn-default dropdown-toggle"
data-toggle="dropdown"
        aria-haspopup="true" aria-expanded="false">
      具有选项分隔线的下拉式菜单
      <span class="caret"></span>
    </button>

    <ul class="dropdown-menu">
```

```
        <li><a href="#">选项 9</a></li>
        <li><a href="#">选项 10</a></li>
        <li><a href="#">选项 11</a></li>
        <li role="separator" class="divider"></li>
        <li><a href="#">选项 12</a></li>
      </ul>
</div>
```

在上述程序代码当中，ul 元素以菜单的模样显示出来，而其内部的 li 元素在此则是表示菜单中的某个选项。在 li 元素中，还有表示**超链接**的 a 元素。

默认的超链接，浏览器会加上**底线**显示出来，当鼠标指针位于超链接的范围内时，会变成**手掌**的形状，以暗示用户是可以选择的。在此的超链接没有显示出**底线**，是因为受到 Bootstrap 的作用。

在此菜单中的分隔线是由 li 元素所构成的，其实也算是菜单中的一个**选项**，受到 role="separator" class="divider" 语句的作用而被应用了 Bootstrap 的分隔线样式。

4.3.4 具有选项分类和选项分隔线的下拉式菜单

比起前一小节的菜单组件，从图 4-4 可以看出此菜单多了**分类标题**和**被停用**的选项。分类标题看起来是被刷淡的文字，而被停用的选项除了会使得范围内的鼠标指针变成**禁用符号**之外，用户也无法选择那个选项上的超链接。

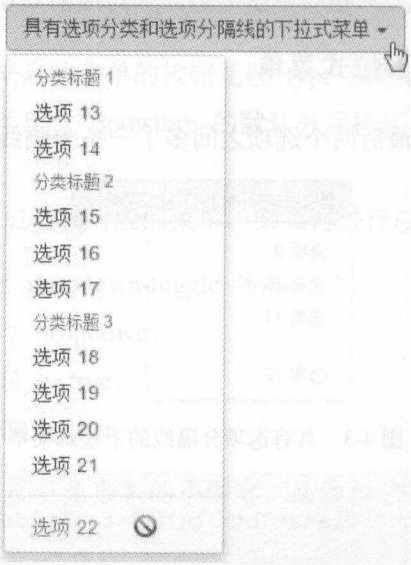


图 4-4 具有选项分类和选项分隔线的下拉式菜单

```
<div class="dropdown">
   <button class="btn btn-default dropdown-toggle" type="button"
      id="dropdownMenu1" data-toggle="dropdown" aria-haspopup="true"
      aria-expanded="true">
```

```
    具有选项分类和选项分隔线的下拉式菜单
    <span class="caret"></span>
  </button>

  <ul class="dropdown-menu" aria-labelledby="dropdownMenu1">
    <li class="dropdown-header">分类标题 1</li>
    <li><a href="#">选项 13</a></li>
    <li><a href="#">选项 14</a></li>
    <li class="dropdown-header">分类标题 2</li>
    <li><a href="#">选项 15</a></li>
    <li><a href="#">选项 16</a></li>
    <li><a href="#">选项 17</a></li>
    <li class="dropdown-header">分类标题 3</li>
    <li><a href="#">选项 18</a></li>
    <li><a href="#">选项 19</a></li>
    <li><a href="#">选项 20</a></li>
    <li><a href="#">选项 21</a></li>
    <li role="separator" class="divider"></li>
    <li class="disabled"><a href="#">选项 22</a></li>
  </ul>
</div>
```

上述菜单主要由如下元素所构成。

- 第一层为 ul 元素，其 class 属性值为 dropdown-menu，aria-labelledby 属性值为上述按钮的 id。
- 第二层为性质不同的多个 li 元素。

上述菜单的**一般**选项之间，有些是紧接在一起的，有些隔着分类标题，有些则是存在分隔线。上述菜单的最后一个**被停用**的选项，当用户将鼠标指针移过去时，就会出现一个禁止图案。上述菜单主要包含以下类型的选项。

- 一般选项

```
<li><a href="#">选项 XX</a></li>
```

- 分类标题

```
<li class="dropdown-header">分类标题 X</li>
```

- 分隔线

```
<li role="separator" class="divider"></li>
```

- 被停用的选项

```
<li class="disabled"><a href="#">选项 XX</a></li>
```

4.4 一般按钮和按钮组

本节的程序代码都来自于 Ch04-04-buttons.html 范例文件。

4.4.1 按钮组

从图 4-5 可以看出，Bootstrap 按钮组是由一组紧密相连的按钮所构成的。

图 4-5 Bootstrap 按钮组

```
<div class="btn-group" role="group" aria-label="...">
   <button type="button" class="btn btn-default">左</button>
   <button type="button" class="btn btn-default">中</button>
   <button type="button" class="btn btn-default">右</button>
</div>
```

在 div 元素中，设置 class 属性值为 btn-group的，role 属性值为 group，安排几个 button 子元素，并将其 type 属性值设置为 button，把 class 属性值设置为 btn btn-default，即可获得一个上述的按钮组组件。

4.4.2 按钮组的工具栏

Bootstrap 工具栏其实是由多个**按钮组**所构成的。从图 4-6 可以看出，笔者刻意让此工具栏按钮组的尺寸变得不同，以便让读者可充分了解其语句的细节。

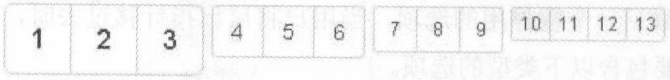


图 4-6 不同尺寸的按钮组

```
<div class="btn-toolbar" role="toolbar" aria-label="...">
   <div class="btn-group btn-group-lg" role="group" aria-label="...">
      <button type="button" class="btn btn-default">1</button>
      <button type="button" class="btn btn-default">2</button>
      <button type="button" class="btn btn-default">3</button>
   </div>

   <div class="btn-group" role="group" aria-label="...">
      <button type="button" class="btn btn-default">4</button>
      <button type="button" class="btn btn-default">5</button>
      <button type="button" class="btn btn-default">6</button>
   </div>

   <div class="btn-group btn-group-sm" role="group" aria-label="...">
      <button type="button" class="btn btn-default">7</button>
```

```
        <button type="button" class="btn btn-default">8</button>
        <button type="button" class="btn btn-default">9</button>
    </div>

    <div class="btn-group btn-group-xs" role="group" aria-label="...">
        <button type="button" class="btn btn-default">10</button>
        <button type="button" class="btn btn-default">11</button>
        <button type="button" class="btn btn-default">12</button>
        <button type="button" class="btn btn-default">13</button>
    </div>
</div>
```

上述的按钮组工具栏主要由如下元素所构成。

- 在 div 元素中，设置 class 属性值为 btn-toolbar，role 属性值为 toolbar。
- 在上述 div 元素中，安排几个第二层的 div 子元素，设置 class 属性值为 btn-group，并加设 btn-group-lg、btn-group-sm 或 btn-group-xs，使得此按钮组在外观上看起来分别是大尺寸 (lg, large)、小尺寸 (sm, small) 和特小尺寸 (xs, extra small)，把 role 属性值设置为 group。
- 在上述第二层的 div 子元素中，安排 button 元素，并把其 type 属性值设置为 button，把 class 属性值设置为 btn btn-default。

4.5 带有菜单按钮的按钮组

本节的程序代码都来自于 Ch04-05-buttons-with-menu.html 范例文件。

4.5.1 横向按钮组

可以看出图 4-7 中的**按钮组**里，带有一个可启动菜单的按钮。因为此菜单按钮被放在**按钮组**中，所以本小节的范例程序代码包含**两个层次**的按钮组结构。

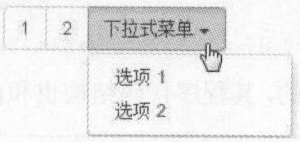


图 4-7 具有两个层次的横向按钮组结构

```
<div class="btn-group" role="group" aria-label="...">
  <button type="button" class="btn btn-default">1</button>
  <button type="button" class="btn btn-default">2</button>

  <div class="btn-group" role="group">
  <button type="button" class="btn btn-default dropdown-toggle"
     data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    下拉式菜单
```

```
      <span class="caret"></span>
    </button>

    <ul class="dropdown-menu">
       <li><a href="#">选项 1</a></li>
       <li><a href="#">选项 2</a></li>
    </ul>
    </div>
</div>
```

带有菜单按钮的横向按钮组，主要由如下元素所构成。

- 在 div 元素中，设置 class 属性值为 btn-group，role 属性值为 group。
- 在上述 div 元素中，再安排几个 button 元素，并把其 type 属性值设置为 button，把 class 属性值设置为 btn btn-default。
- 在上述 div 元素中，再安排至少一个第二层的 div 子元素，并把 class 属性值设置为 btn-group，把 role 属性值设置为 group 。

在上述第二层 div 子元素中：

- 安排一个 button 元素，并设置 type 属性值为 button，设置 class 属性值为 btn btn-default dropdown-toggle，使得它拥有下拉式菜单按钮的外观！其中：
  - data-toggle="dropdown"
  - aria-haspopup="true"
  - aria-expanded="false"
- 安排一个设置 class 属性值为 caret 的 span 子元素，使得出现**倒三角形**小图标。
- 安排一个 ul 元素，并设置 class 属性值为 dropdown-menu，使得该 ul 元素成为菜单组件！

4.5.2 纵向按钮组

图 4-8 中所示的按钮组是**纵向**的，其程序代码结构也和前一小节提及的横向按钮组几乎是一模一样的。

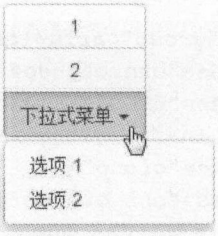


图 4-8　纵向按钮组

```
<div class="btn-group-vertical" role="group" aria-label="...">
  <button type="button" class="btn btn-default">1</button>
  <button type="button" class="btn btn-default">2</button>

  <div class="btn-group" role="group">
    <button type="button" class="btn btn-default dropdown-toggle"
       data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
      下拉式菜单
      <span class="caret"></span>
    </button>

    <ul class="dropdown-menu">
      <li><a href="#">选项 1</a></li>
      <li><a href="#">选项 2</a></li>
    </ul>
  </div>
</div>
```

和前一小节所提及的**横向**按钮组相比，只要在此组件第一层的div元素中设置 class 属性值为 btn-group-**vertical**，设置role 属性值为 group，即可将**横向**的按钮组变成**纵向**的按钮组了。

4.5.3 窗口宽度型

图 4-9 中所示的按钮组会横跨整个浏览器窗口宽度，而且在默认情况下，内部每个按钮的宽度是相同的。因此，若在较小型的移动设备屏幕上，此按钮组会显示得比较窄，内部每个按钮自然也会比较小。

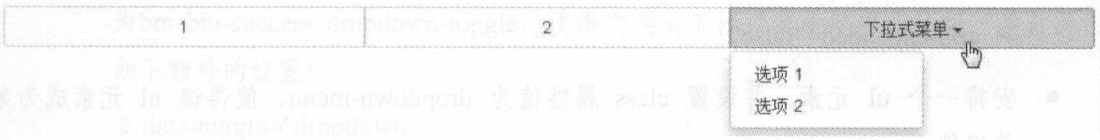


图 4-9 窗口宽度型按钮组

```
<div class="btn-group btn-group-justified" role="group" aria-label="...">
  <div class="btn-group" role="group">
    <button type="button" class="btn btn-default">1</button>
  </div>

  <div class="btn-group" role="group">
    <button type="button" class="btn btn-default">2</button>
  </div>

  <div class="btn-group" role="group">
    <button type="button" class="btn btn-default dropdown-toggle"
       data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
      下拉式菜单
      <span class="caret"></span>
    </button>
```

```
      <ul class="dropdown-menu">
        <li><a href="#">选项 1</a></li>
        <li><a href="#">选项 2</a></li>
      </ul>
    </div>
</div>
```

窗口宽度型按钮组主要由如下元素所构成。

- 在 div 元素中，设置 class 属性值为 btn-group btn-group-justified，设置 role 属性值为 group。
- 在上述的 div 元素中，再安排几个第二层的 div 子元素，并把 class 属性值设置为 btn-group，把 role 属性值设置为 group。

在上述第二层 div 子元素中：

- 有些安排了一个 button 元素，并把其 type 属性值设置为 button，把 class 属性值设置为 btn btn-default。
- 有些则安排了一个 button 元素，把其 type 属性值设置为 button，把 class 属性值设置为 btn btn-default dropdown-toggle，使得它拥有下拉式菜单按钮的外观，额外设置如下语句：
  - data-toggle="dropdown"
  - aria-haspopup="true"
  - aria-expanded="false"
- 安排一个 ul 元素，并设置 class 属性值为 dropdown-menu，使得该 ul 元素成为菜单组件。

4.5.4 分离型按钮组

如图 4-10 所示的菜单按钮是由左侧较宽的按钮和右侧包含**倒三角**图案的较窄按钮共同组成的。因此，程序设计人员可以给左侧和右侧按钮赋予不同的功能。

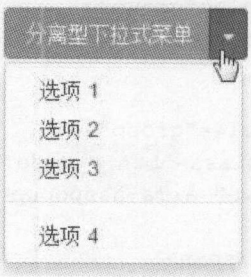


图 4-10 分离型按钮组

```
<div class="btn-group">
  <button type="button" class="btn btn-success">分离型下拉式菜单</button>
  <button type="button" class="btn btn-success dropdown-toggle"
     data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    <span class="caret"></span>
  </button>
  <ul class="dropdown-menu">
    <li><a href="#">选项 1</a></li>
    <li><a href="#">选项 2</a></li>
    <li><a href="#">选项 3</a></li>
    <li role="separator" class="divider"></li>
    <li><a href="#">选项 4</a></li>
  </ul>
</div>
```

分离型下拉式菜单按钮组是由两个 button 按钮元素共同构成的。在左侧按钮元素的外观上可显示文字，而在右侧按钮元素的外观上可用来显示下拉或上拉菜单的**倒三角形**符号。

分离型下拉式菜单按钮组主要由如下元素所构成。

- 在 div 元素中，设置 class 属性值为 btn-group。
- 在上述 div 元素中：
  - 先安排一个button 按钮元素，并设置 class 属性值为 btn btn-success。其中，btn-success 可使得此按钮元素的外观显示为绿色。
  - 再安排另一个 button 按钮元素，设置其 type 属性值为 button，设置class 属性值为btn btn-success dropdown-toggle，使得它拥有下拉式菜单按钮的外观，并且进行如下额外的设置：
    - ◊ data-toggle="dropdown"
    - ◊ aria-haspopup="true"
    - ◊ aria-expanded="false"
  - 最后安排一个 ul 元素，并设置 class 属性值为 dropdown-menu，使得该 ul 元素成为菜单组件。

4.6 不同尺寸的按钮

本节的程序代码都来自于 Ch04-06-scaled-buttons.html 范例文件。

4.6.1 大尺寸按钮

图 4-11 中的菜单按钮是大尺寸的，所以在不同移动设备上被显示时，总会比默认的按钮尺寸大。

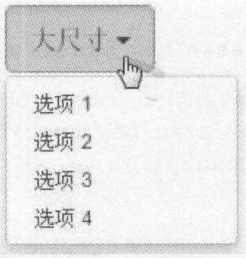


图 4-11　大尺寸的菜单按钮

```
<div class="btn-group">
   <button class="btn btn-default btn-lg dropdown-toggle" type="button"
       data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
     大尺寸
     <span class="caret"></span>
   </button>

   <ul class="dropdown-menu" aria-labelledby="dropdownMenu1">
     <li><a href="#">选项 1</a></li>
     <li><a href="#">选项 2</a></li>
     <li><a href="#">选项 3</a></li>
     <li><a href="#">选项 4</a></li>
   </ul>
</div>
```

为了得到**大**尺寸的下拉式菜单按钮，可如下设置相关元素。

- 在 div 元素中，设置 class 属性值为 btn-group。
- 在上述 div 元素当中：
  - ➢ 安排一个 button 按钮元素，设置其 class 属性值为 btn btn-default btn-lg dropdown-toggle，其中 btn-lg可使得此按钮元素的外观显示成大尺寸，并设置其 type 属性值为 button，以及：
    - ◊ data-toggle="dropdown"
    - ◊ aria-haspopup="true"
    - ◊ aria-expanded="false"
  - ➢ 安排一个 ul 元素，并设置 class 属性值为 dropdown-menu，使得该 ul 元素成为菜单组件！

4.6.2　标准尺寸的上拉式菜单按钮

图 4-12 中的菜单按钮是**标准**尺寸的。

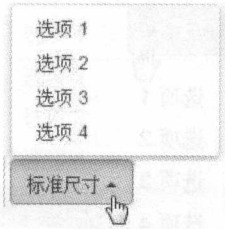


图 4-12 标准尺寸的菜单按钮

```
    <div class="btn-group dropup">
      <button class="btn btn-default dropdown-toggle" type="button"
data-toggle="dropdown"
        aria-haspopup="true" aria-expanded="false">
        标准尺寸
        <span class="caret"></span>
      </button>

      <ul class="dropdown-menu" aria-labelledby="dropdownMenu1">
        <li><a href="#">选项 1</a></li>
        <li><a href="#">选项 2</a></li>
        <li><a href="#">选项 3</a></li>
        <li><a href="#">选项 4</a></li>
      </ul>
    </div>
```

为了得到**标准**尺寸的**上拉式**菜单按钮，可如下设置相关元素。

- 在 div 元素中，设置 class 属性值为 btn-group dropup，其中 dropup 可使得菜单变成上拉式的。
- 在上述 div 元素当中：
  - 安排一个 button 按钮元素，设置其 class 属性值为 btn btn-default dropdown-toggle，只要不带 btn-lg、btn-sm 或 btn-xs 类型的CSS 规则名称，即可使得此按钮元素的外观显示成标准尺寸。设置其 type 属性值为 button，以及：
    - ◊ data-toggle="dropdown"
    - ◊ aria-haspopup="true"
    - ◊ aria-expanded="false"
  - 安排一个 ul 元素，并设置 class 属性值为 dropdown-menu，使得该 ul 元素成为菜单组件。

4.6.3 小尺寸按钮

图 4-13 中的菜单按钮是**小**尺寸的，所以在不同移动设备上被显示时，总会比默认的按钮尺寸**小**。

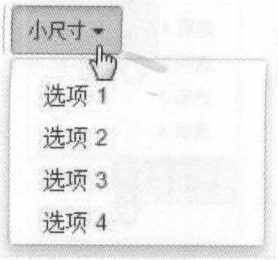


图 4-13　小尺寸的菜单按钮

```
<div class="btn-group">
  <button class="btn btn-default btn-sm dropdown-toggle" type="button"
      data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    小尺寸
    <span class="caret"></span>
  </button>

  <ul class="dropdown-menu" aria-labelledby="dropdownMenu1">
    <li><a href="#">选项 1</a></li>
    <li><a href="#">选项 2</a></li>
    <li><a href="#">选项 3</a></li>
    <li><a href="#">选项 4</a></li>
  </ul>
</div>
```

为了得到**小**尺寸的下拉式菜单按钮，可如下设置各元素。

- 在 div 元素中，设置 class 属性值为 btn-group。
- 在上述 div 元素当中：
  - 安排一个 button 按钮元素，设置其 class 属性值为 btn btn-default btn-sm dropdown-toggle，其中 btn-sm 可使得此按钮元素的外观显示成小尺寸，并设置其 type 属性值为 button，以及：
    - ◊ data-toggle="dropdown"
    - ◊ aria-haspopup="true"
    - ◊ aria-expanded="false"
  - 安排一个 ul 元素，并设置 class 属性值为 dropdown-menu，使得该 ul 元素成为菜单组件。

4.6.4　特小尺寸按钮

图 4-14 中的菜单按钮是**特小**尺寸的，所以在不同移动设备上被显示时，总会比默认的按钮尺寸显得**特别小**。

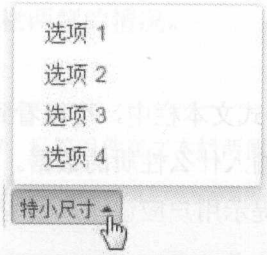


图 4-14 特小尺寸的菜单按钮

```
<div class="btn-group dropup">
  <button class="btn btn-default btn-xs dropdown-toggle" type="button"
     data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
    特小尺寸
    <span class="caret"></span>
  </button>

  <ul class="dropdown-menu" aria-labelledby="dropdownMenu1">
    <li><a href="#">选项 1</a></li>
    <li><a href="#">选项 2</a></li>
    <li><a href="#">选项 3</a></li>
    <li><a href="#">选项 4</a></li>
  </ul>
</div>
```

为了得到**特小**尺寸的菜单按钮，可如下设置相关程序代码。

- 在 div 元素中，设置 class 属性值为 btn-group。
- 在上述 div 元素当中：
  - 安排一个 button 按钮元素，设置其 class 属性值为 btn btn-default btn-xs dropdown-toggle，其中 btn-xs可使得此按钮元素的外观显示成特小尺寸，并设置其 type 属性值为 button，以及：
    - ◊ data-toggle="dropdown"
    - ◊ aria-haspopup="true"
    - ◊ aria-expanded="false"
  - 安排一个 ul 元素，并设置 class 属性值为 dropdown-menu，使得该 ul 元素成为菜单组件。

4.7 窗体中的文本栏

本节的程序代码都来自于 Ch04-07-form-textfields.html 范例文件。

4.7.1 标签文字组合式

在图 4-15 所示的标签文字组合式文本栏中，可以看到在文本栏的左侧有一个标签组件，用来告知用户其右侧的文本栏应该输入什么性质的数据。

而文本栏会显示浅灰色字体，提示用户应该输入哪些字符作为数据内容。

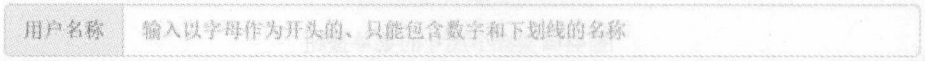


图 4-15 标签文字组合式文本栏

```
<div class="input-group input-group-lg">
  <span class="input-group-addon" id="basic-addon1">用户名称</span>
  <input type="text" class="form-control" placeholder="输入以字母作为开头的、
只能包含数字和下划线的名称" aria-describedby="basic-addon1">
</div>
```

为了得到如上的标签文字组合式文本栏，可如下设置相关程序代码。

- 在第一个 div 元素中，设置 class 属性值为 input-group input-group-lg，并且：
  - 安排一个 span 元素，设置其 class 属性值为 input-group-addon，其内容区为标签文字（例如：用户名称）。
  - 安排一个 input 元素，设置其 type 属性值为 text，设置其 class 属性值为 form-control，而设置其 placeholder 属性值为要在文本栏中显示给用户看的提示文字。

在图 4-16 中，可以看出标签组件被移至文本栏的右侧。

图 4-16 标签组件移至文本栏右侧的情况

```
<p style="height: 20px"></p>

<div class="input-group">
  <input type="text" class="form-control" placeholder="Email 账号名称"
    aria-describedby="basic-addon2">
  <span class="input-group-addon"
    id="basic-addon2">@world.galaxy.com</span>
</div>
```

- 在第二个 div 元素中，设置 class 属性值为 input-group，并且：
  - 安排一个 input 元素，设置其 type 属性值为 text，其 class 属性值为 form-control，其 placeholder 属性值为要在文本栏中显示给用户看的提示文字。
  - 安排一个 span 元素，设置其 class 属性值为input-group-addon，其内容区为标签文字（例如：@world.galaxy.com）。

图 4-17 所示为标签组件在文本栏两侧的情况。

图 4-17 标签组件在文本栏两侧的情况

```
<p style="height: 20px"></p>

<div class="input-group input-group-sm">
   <span class="input-group-addon">结账金额</span>

   <input type="text" class="form-control" placeholder="请输入整数值">

   <span class="input-group-addon">元</span>
</div>
```

- 在第三个 div 元素中，设置 class 属性值为 input-group input-group-sm，其中 input-group-sm 会使得此输入群组的外观看起来较小，并且：
  - 安排一个 input 元素，设置其 type 属性值为 text，其 class 属性值为 form-control，其 placeholder 属性值为要在文本栏中显示给用户看的提示文字。
  - 在上述 input 元素的两旁各安排一个 span 元素，设置其 class 属性值为 input-group-addon，其内容区为标签文字（例如：结账金额，元）。

在本小节的范例中，无论是 input 元素或是 span 元素，都是所谓的**行内**元素。因此在默认情况下，浏览器会将它们从左到右地横向排列，直到剩余宽度不足以显示后续的**行内**元素，才会从下一行继续显示剩余的元素。

4.7.2 单选按钮和多选按钮组合式

在图 4-18 中，浏览器默认会在**同一行**中，其**左侧**显示**多选按钮**（复选框）和**文本栏**的组合，在其**右侧**显示**单选按钮**和**文本栏**的组合。

图 4-18 在同一行显示的多选按钮和文本栏的组合与单选按钮和文本栏的组合

当窗口宽度不足时，浏览器会自动改为如图 4-19 所示的外观显示。

图 4-19 分行显示的多选按钮和文本栏的组合与单选按钮和文本栏的组合

从图 4-19 可以看到，多选按钮和文本栏的组合与单选按钮和文本栏的组合变成纵向排列了。

```
<div class="row">
 <div class="col-lg-6">
  <div class="input-group">
   <span class="input-group-addon">
    <input type="checkbox" aria-label="...">
   </span>

   <input type="text" class="form-control" aria-label="..." placeholder="
可输入文字...">
  </div><!-- /input-group -->
 </div><!-- /.col-lg-6 -->

 <div class="col-lg-6">
  <div class="input-group">
   <span class="input-group-addon">
    <input type="radio" aria-label="...">
   </span>

   <input type="text" class="form-control" aria-label="..." placeholder="
可输入文字...">
  </div><!-- /input-group -->
 </div><!-- /.col-lg-6 -->
</div><!-- /.row -->
```

从如上程序代码可以看到，总共存在 2 组**从外到内**嵌套了 3 层 div 元素的结构，从外层到内层的各 div 元素的 class 属性值分别是 row、col-lg-6 和 input-group，使得该层 div 元素所显示的外观为下列各组件。

- row 表示**行集合**组件。
- col-lg-6 表示此元素在大尺寸（lg, large）窗口宽度时会成为占用 12 等分中 6 等分的**版型组件**，也就是占用**窗口宽度**的一半（6 / 12）。
- input-group 表示**输入群组**组件。

最内层 div 元素的内部尚有其他 span、input 元素，以构成多选按钮（type="checkbox"）、单选按钮（type="radio"）的组合式文本栏。

在本范例中，无论是文本栏、单选按钮或多选按钮，其 HTML 的语句都是由 input 元素所构成的，只不过其 type 属性值分别为 text、radio 和 checkbox，从而有所不同。

4.7.3 一般按钮组合式

在图 4-20 中，左侧是**按钮**与**文本栏**的组合，而其右侧是先**文本栏**再**按钮**的组合。

图 4-20 一般按钮组合式

当窗口宽度不足时，浏览器会自动改为如图 4-21 所示的外观显示。

图 4-21 当窗口宽度不足时一般按钮组合式的外观显示

从图 4-21 可以看到，按钮和文本栏的组合与文本栏和按钮的组合变成纵向排列了。

```
<div class="row">
 <div class="col-lg-6">
   <div class="input-group">
     <span class="input-group-btn">
       <button class="btn btn-default" type="button">搜索</button>
     </span>

     <input type="text" class="form-control" placeholder="请输入要搜索的关键字">
   </div><!-- /input-group -->
 </div><!-- /.col-lg-6 -->

 <div class="col-lg-6">
   <div class="input-group">
     <input type="text" class="form-control" placeholder="请输入要搜索的关键字">

     <span class="input-group-btn">
       <button class="btn btn-default" type="button">搜索</button>
     </span>
   </div><!-- /input-group -->
 </div><!-- /.col-lg-6 -->
</div><!-- /.row -->
```

如上的程序代码和前一小节的程序代码很类似，只不过前一小节放入的是多选按钮或单选按钮，而本程序代码放入的则是 class 属性值为 btn btn-default 的 button（按钮）元素，以及 class 属性值为 input-group-btn 的 span 元素。

4.7.4 下拉式菜单按钮组合式

从图 4-22 中可以看到下拉式菜单按钮和文本栏的组合。

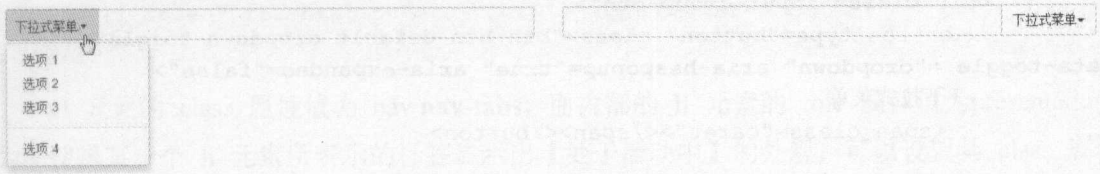


图 4-22 下拉式菜单按钮组合式

当窗口宽度不足时，浏览器会自动改为如图 4-23 所示的外观显示。

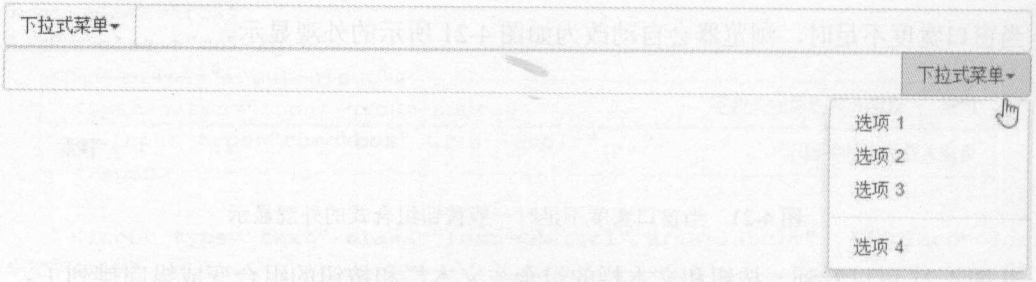


图 4-23　当浏览器窗口宽度不足时下拉式菜单按钮组合的显示方式

从图 4-23 可以看到，下拉式菜单按钮和文本栏的组合与文本栏和下拉式菜单按钮的组合变成纵向排列了。

```
    <div class="row">
     <div class="col-lg-6">
       <div class="input-group">
         <div class="input-group-btn">
           <button type="button" class="btn btn-default dropdown-toggle"
data-toggle="dropdown" aria-haspopup="true" aria-expanded="false">
            下拉式菜单
            <span class="caret"></span></button>

          <ul class="dropdown-menu">
            <li><a href="#">选项 1</a></li>
            <li><a href="#">选项 2</a></li>
            <li><a href="#">选项 3</a></li>
            <li role="separator" class="divider"></li>
            <li><a href="#">选项 4</a></li>
          </ul>
        </div><!-- /btn-group -->

        <input type="text" class="form-control" aria-label="...">
      </div><!-- /input-group -->
     </div><!-- /.col-lg-6 -->

     <div class="col-lg-6">
       <div class="input-group">
         <input type="text" class="form-control" aria-label="...">

         <div class="input-group-btn">
           <button type="button" class="btn btn-default dropdown-toggle"
data-toggle ="dropdown" aria-haspopup="true" aria-expanded="false">
            下拉式菜单
            <span class="caret"></span></button>

          <ul class="dropdown-menu dropdown-menu-right">
            <li><a href="#">选项 1</a></li>
            <li><a href="#">选项 2</a></li>
            <li><a href="#">选项 3</a></li>
```

```
            <li role="separator" class="divider"></li>
            <li><a href="#">选项 4</a></li>
          </ul>
        </div><!-- /btn-group -->
      </div><!-- /input-group -->
    </div><!-- /.col-lg-6 -->
  </div><!-- /.row -->
```

和前小节的程序代码相似，但放入的是具有如下属性的下拉式菜单按钮。

- type 属性值为 button。
- class 属性值为 btn btn-default dropdown-toggle。
- data-toggle 属性值为 dropdown。
- aria-haspopup 属性值为 true。
- aria-expanded 属性值为 false。

ul 元素的 class 属性值若被加注 dropdown-menu-right，可使得自身成为在浏览器窗口**右侧**下拉显示的菜单。

4.8 窗体中的导航栏

本节的程序代码都来自于 Ch04-08-form-navigators.html 范例文件。

4.8.1 默认的标签型导航栏

图 4-24 所示是一个默认的**标签型导航栏**组件，它是由多个标签项所组成的，在此分别为**首页**、**简介**和**联络方式**。

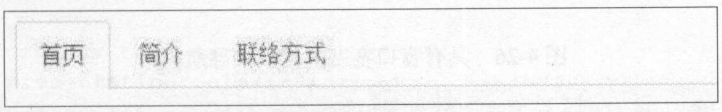


图 4-24 标签型导航栏

```
<ul class="nav nav-tabs">
  <li role="presentation" class="active"><a href="#">首页</a></li>
  <li role="presentation"><a href="#">简介</a></li>
  <li role="presentation"><a href="#">联络方式</a></li>
</ul>
```

ul 元素的 class 属性值为 nav **nav-tabs**，而内部的 li 元素的 role 属性值为presentation。为了使得某一个 li 元素所表示的标签显示出【处于活动中】的外观，可以设置其 class 属性值为 active。

4.8.2 简易标签型导航栏

和前一小节的范例相比，在图 4-25 所示的标签型导航栏中少了标签的**边框**和导航栏的**底线**，但是标签上多了**蓝色背景**，表示这个标签处于“活动”状态中，而其**文字**会显示**白色**。

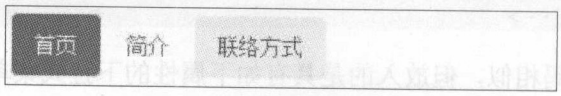


图 4-25 简易型导航栏

```
<ul class="nav nav-pills">
   <li role="presentation" class="active"><a href="#">首页</a></li>
   <li role="presentation"><a href="#">简介</a></li>
   <li role="presentation"><a href="#">联络方式</a></li>
</ul>
```

ul 元素的 class 属性值为 nav **nav-pills**，而内部的 li 元素的 role 属性值为 presentation。为了使某一个 li 元素所表示的标签显示出【处于活动中】的外观，可以设置其 class 属性值为 active。

4.8.3 具有窗口宽度的堆栈式导航栏

在图 4-26 所示的具有窗口宽度的堆栈式导航栏中，各个按钮的宽度是窗口的宽度，并且以**纵向**排列。

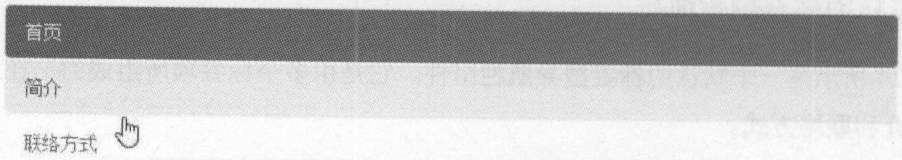


图 4-26 具有窗口宽度的堆栈式导航栏

```
<ul class="nav nav-pills nav-stacked">
   <li role="presentation" class="active"><a href="#">首页</a></li>
   <li role="presentation"><a href="#">简介</a></li>
   <li role="presentation"><a href="#">联络方式</a></li>
</ul>
```

ul 元素的 class 属性值为 nav nav-pills **nav-stacked**，而内部的 li 元素的 role 属性值为 presentation。为了使某一个 li 元素所表示的标签显示出【处于活动中】的外观，可以设置其 class 属性值为 active。

4.8.4 均分窗口宽度的标签型导航栏

从图 4-27 可以看到有 3 组均分窗口宽度的标签型导航栏。在第 2 组的最后一个标签是【被停用中】的标签，不仅以浅灰色显示，当鼠标指针落在其范围内时，就会变成**禁用**符号。

在第 3 组的最后一个标签实际上是一个下拉式菜单按钮，可供单击以显示一个下拉式菜单。

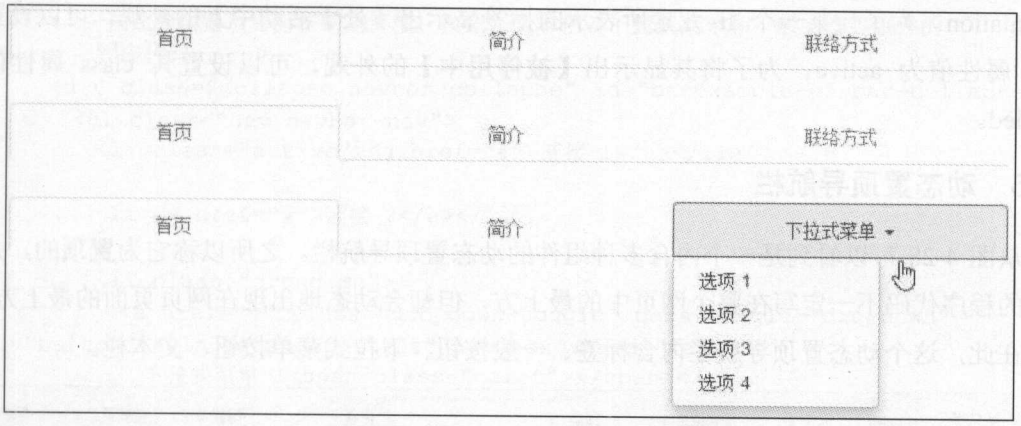


图 4-27 均分窗口宽度的标签型导航栏

```
<ul class="nav nav-tabs nav-justified">
   <li role="presentation" class="active"><a href="#">首页</a></li>
   <li role="presentation"><a href="#">简介</a></li>
   <li role="presentation"><a href="#">联络方式</a></li>
</ul>

<p style="height: 20px"></p>

<ul class="nav nav-tabs nav-justified">
 <li role="presentation" class="active"><a href="#">首页</a></li>
 <li role="presentation"><a href="#">简介</a></li>
 <li role="presentation" class="disabled"><a href="#">联络方式</a></li>
</ul>

<p style="height: 20px"></p>

<ul class="nav nav-tabs nav-justified">
 <li role="presentation" class="active"><a href="#">首页</a></li>
 <li role="presentation"><a href="#">简介</a></li>
 <li role="presentation" class="dropdown">
   <a class="dropdown-toggle" data-toggle="dropdown" href="#" role="button"
aria-haspopup="true" aria-expanded="false">
     下拉式菜单<span class="caret"></span>
   </a>

   <ul class="dropdown-menu">
     <li><a href="#">选项 1</a></li>
     <li><a href="#">选项 2</a></li>
     <li><a href="#">选项 3</a></li>
     <li><a href="#">选项 4</a></li>
   </ul>
 </li>
</ul>
```

ul 元素的 class 属性值为 nav nav-tabs **nav-justified**，而内部的 li 元素的 role 属性值为 presentation。为了让某一个 li 元素所表示的标签显示出**【处于活动中】**的外观，可以设置其 class 属性值为 active；为了将其显示出**【被停用中】**的外观，可以设置其 class 属性值为 disabled。

4.8.5 动态置顶导航栏

从图 4-28 可以看到是一个内含多种组件的动态置顶导航栏。之所以称它为**置顶**的，是因为它的程序代码不一定写在整个网页中的最上方，但却会动态地出现在网页页面的最上方。

在此，这个动态置顶导航栏内含标签、一般按钮、下拉式菜单按钮、文本栏。

图 4-28　动态置顶导航栏

当窗口宽度不足时，浏览器会自动改为如图 4-29 所示的外观显示。

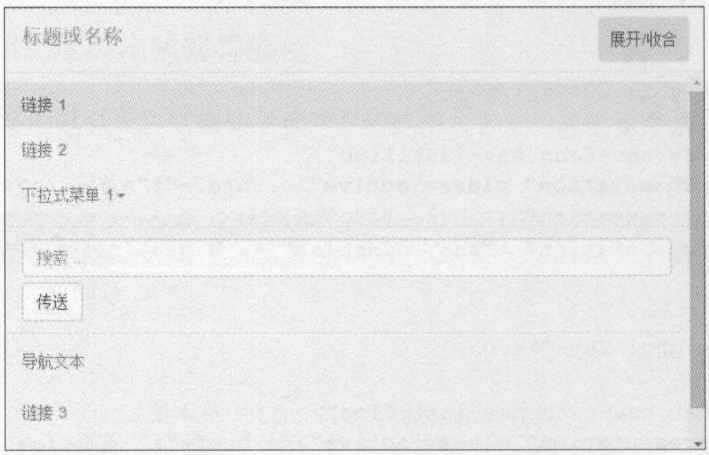


图 4-29　当窗口宽度不足时动态置顶导航栏的外观显示

```
    <!--需调整 body 元素上方填充距离 padding-top 的值，以避免网页上其他内容和如下
        【置顶】导航栏重叠在一起。 -->
    <nav class="navbar navbar-default navbar-fixed-top">
    <!--<nav class="navbar navbar-default navbar-fixed-bottom"> -->
      <div class="container-fluid">
        <div class="navbar-header">
          <button type="button" class="navbar-toggle collapsed" data-toggle=
"collapse" data-
    target="#bs-example-navbar-collapse-1" aria-expanded="false">展开/收合
</button>

          <a class="navbar-brand" href="#">标题或名称</a>
```

```
    </div>

    <!--在此列出置顶导航栏的超链接、按钮、文本栏、下拉式菜单和一般文本等组件的
        程序代码。 -->
    <div class="collapse navbar-collapse" id="bs-example-navbar-collapse-1">
      <ul class="nav navbar-nav">
        <li class="active"><a href="#">链接 1</a></li>

        <li><a href="#">链接 2</a></li>

        <li class="dropdown">
          <a href="#" class="dropdown-toggle" data-toggle="dropdown"
role="button" aria-haspopup="true" aria-expanded="false">
            下拉式菜单 1<span class="caret"></span></a>

          <ul class="dropdown-menu">
            <li><a href="#">选项 1</a></li>
            <li><a href="#">选项 2</a></li>
            <li><a href="#">选项 3</a></li>
            <li role="separator" class="divider"></li>
            <li><a href="#">选项 4</a></li>
            <li role="separator" class="divider"></li>
            <li><a href="#">选项 5</a></li>
          </ul>
        </li>
      </ul>

      <form class="navbar-form navbar-left" role="search">
        <div class="form-group">
          <input type="text" class="form-control" placeholder="搜索 ...">
        </div>

        <button type="submit" class="btn btn-default">提交</button>
      </form>

      <p class="navbar-text navbar-right">导航文本</p>

      <ul class="nav navbar-nav navbar-right">
        <li><a href="#">链接 3</a></li>

        <li class="dropdown">
          <a href="#" class="dropdown-toggle" data-toggle="dropdown"
role="button" aria-haspopup="true" aria-expanded="false">
            下拉式菜单 2
            <span class="caret"></span>
          </a>

          <ul class="dropdown-menu">
            <li><a href="#">选项 1</a></li>
            <li><a href="#">选项 2</a></li>
```

```
            <li><a href="#">选项 3</a></li>
            <li role="separator" class="divider"></li>
            <li><a href="#">选项 4</a></li>
          </ul>
        </li>
      </ul>
    </div><!-- /.navbar-collapse -->
  </div><!-- /.container-fluid -->
</nav>
```

设置 nav 元素的 class 属性值为 navbar navbar-default navbar-fixed-top，其中 navbar-fixed-top 会使得其导航栏显示成【**置顶**】的样子；而 navbar-fixed-bottom 则会使得其导航栏显示【**置底**】的样子。

在 class 属性值为 container-fluid 的 div 元素就是构成其动态【**置顶 / 置底**】导航栏的各部分结构。其中：

- class 属性值为 navbar-header 的 div 元素中内含【展开 / 收合】按钮以及导航栏的标题或名称。
- class 属性值为 collapse navbar-collapse 的 div 元素中内含超链接、按钮、文本栏、下拉式菜单和一般文本等组件的程序代码。

4.9 浏览分层提示和分页

本节的程序代码都来自于 Ch04-09-breadcrumbs-and-paginations.html 范例文件。

4.9.1 浏览分层提示

如图 4-30 所示，可以看到所谓的浏览分层提示，其实就是提示用户已浏览到哪个路径上的页面了，在此为【分公司】这个页面上，所以此页面所对应的项为浅灰色字样，表示无法使用鼠标指针来选择的状态。

若要回到前一页面，可用鼠标指针选择路径上前一页面的超链接，例如：【服务网点】。

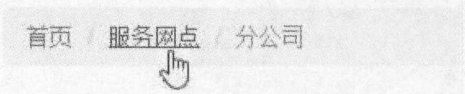


图 4-30 浏览分层提示

```
<ol class="breadcrumb">
  <li><a href="#">首页</a></li>
  <li><a href="#">服务网点</a></li>
  <li class="active">分公司</li>
</ol>
```

通过 class 属性值为 breadcrumb 的 ol 元素和内部的 li 元素来实现浏览分层提示组件。要表示成为【处于活动中】的浏览分层，可设置其 li 元素的 class 属性值为 active。

4.9.2 分页导航栏

图 4-31 所示的分页导航栏是由好几个**超链接**项所构成的：**前一页**、**第 1 页**至**第 5 页**、**下一页**。

图 4-31 分页导航栏

```
<nav>
  <!--<ul class="pagination"> -->
  <ul class="pagination pagination-lg">
  <!--<ul class="pagination pagination-sm"> -->
    <li class="disabled">
      <a href="#" aria-label="Previous">
        <span aria-hidden="true">&laquo;</span>
      </a>
    </li>

    <li><a href="#">1</a></li>
    <li class="active"><a href="#">2</a></li>
    <li><a href="#">3</a></li>
    <li><a href="#">4</a></li>
    <li><a href="#">5</a></li>

    <li>
      <a href="#" aria-label="Next">
        <span aria-hidden="true">&raquo;</span>
      </a>
    </li>
  </ul>
</nav>
```

在 nav 元素中，安排了 class 属性值为 pagination pagination-lg 的 ul 元素，其中 pagination-lg 可使得其分页导航栏显示成大尺寸，而 pagination-sm 可使得其分页导航栏显示为小尺寸。要表示成**【停用中】**的上一页 / 下一页按钮，可在其 a 元素上层的 li 元素语句上设置其 class 属性值为 disabled。要表示成**【处于活动中】**的页，可在其对应的 li 元素语句上设置 class 属性值为 active。

4.9.3 前一页和下一页按钮

在图 4-32 中的 3 组**前一页**和**下一页**按钮中，第 1 组是靠拢的；第 2 组和第 3 组会根据窗口宽度来决定它们之间的距离有多远。

其中，第 3 组的**前一页**按钮处于【**停用中**】状态，不能被鼠标指针选择，以暗示用户当前的**页面**是**第 1 页**，因此不能单击【前一页】按钮。

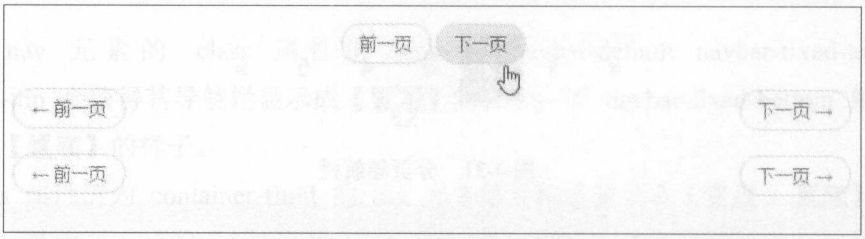


图 4-32 前一页和后一页按钮

```
    <nav>
      <ul class="pager">
        <li><a href="#">前一页</a></li>
        <li><a href="#">下一页</a></li>
      </ul>
    </nav>

    <nav>
      <ul class="pager">
        <li class="previous"><a href="#"><span aria-hidden="true">&larr;</span>
            前一页</a></li>
        <li class="next"><a href="#">下一页<span aria-hidden="true">&rarr;</span>
</a></li>
      </ul>
    </nav>

    <nav>
      <ul class="pager">
        <li class="previous disabled"><a href="#"><span aria-hidden="true">&larr;
</span>
            前一页</a></li>
        <li class="next"><a href="#">下一页<span aria-hidden="true">&rarr;</span>
</a></li>
      </ul>
    </nav>
```

简易型前一页和下一页的超链接按钮可通过在 nav 元素中安排 class 属性值为 pager 的 ul 元素来实现。关于其内的 li 元素，若 class 属性值被设置为 previous，则显示出**前一页**按钮的外观；若 class 属性值被设置为 next，则显示出**下一页**按钮的外观；若 class 属性值增加了 disabled，则会显示【停用中】的状态。

4.10 文字标签、文字徽章与文字框

本节的程序代码都来自于 Ch04-10-labels-badges-and-jumbotron.html 范例文件。

4.10.1 文字标签

图 4-33 中的 6 个文字标签各拥有不同的背景样式，而其字体都是白色粗体字。6 种背景样式分别对应到 Bootstrap 的 6 种 CSS 样式名称：**默认（default）**、**主要（primary）**、**成功（success）**、**信息（info）**、**警告（warning）**与**危险（danger）**。

图 4-33 文字标签

```
<span class="label label-default">默认的颜色</span>
<span class="label label-primary">主要的颜色</span>
<span class="label label-success">成功的颜色</span>
<span class="label label-info">信息的颜色</span>
<span class="label label-warning">警告的颜色</span>
<span class="label label-danger">危险的颜色</span>
```

在 span 元素中，设置其 class 属性值为 label label-default，使其具有文字标签的默认外观。若将 label-default 改为 label-primary、label-success、label-info、label-warning、label-danger，则文字标签会有外观颜色上的变化！

4.10.2 文字徽章

图 4-34 中所示的文字徽章都有一个**标题**或**文字名称**，随后带有一个**内含数值**的**小泡泡框**。在数值为 0 的状态下，可考虑暂时不显示带有数值 0 的小泡泡框，例如【广告邮件】项。

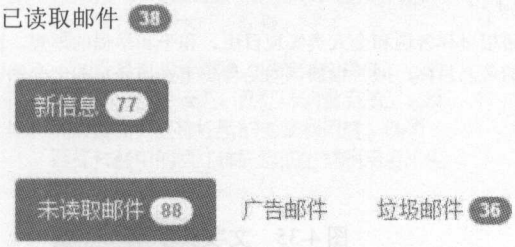


图 4-34 文字徽章中的数值显示方式

```
<a href="#">已读取邮件<span class="badge">38 </span></a>

<p style="margin: 30px"></p>

<button class="btn btn-primary" type="button">新信息<span class="badge">77
</span>
```

图 4-37　在窗口宽度足够时缩略图和图解说明的显示方式

```
    <div class="row">
      <div class="col-md-3">
        <div class="thumbnail">
          <img src="images/Lycopene_powder.jpg" alt="...">

          <div class="caption">
            <h3>(番) 茄红素</h3>

              <p>西红柿红素（Lycopene、分子式 C<sub>40</sub>H<sub>56</sub>）
                 是一种明亮红色的类胡萝卜素颜料，在西红柿和其他红色水果如西瓜
                 和西柚中也有。</p>

              <p>西红柿红素含量高的水果和蔬菜有番茄（容易受成熟度和品种所影响）、
                 西瓜、葡萄柚、番石榴、木瓜、红椒。</p>

              <p>不同于其他的营养素，例如维生素 C，会在烹煮的过程中流失。番茄的
                 食品加工，反而会提高西红柿红素的生物利用度。例如：番茄酱中的
                 西红柿红素的生物利用度比生鲜番茄高了四倍。</p>

              <p><a href="#" class="btn btn-primary" role="button">购买相关产品</a>
</p>
            </div>
          </div>
      </div>
    </div>
```

缩略图与图解说明组件具有较复杂的 4 层 div 元素的程序代码结构，其 class 属性值从外到内分别为：

- row，表示一个**行集合**组件。
- col-md-3，表示在中等尺寸宽度的窗口里，显示出窗口宽度中的 **12 等分**当中的 **3 等分**宽度，也就是**四分之一**的窗口宽度。
- thumbnail，表示缩略图子组件，**其内含设置图像文件的 img 元素**。
- caption，表示图解说明标题子组件，**其内含对应缩略图相关说明的图解说明内容。**

4.12 警告文字框

本节的程序代码都来自于 Ch04-12-alerts.html 范例文件。

在图 4-38 中的警告文字框带有不同颜色样式的矩形框，其中第 1 个警告文字框右侧还带有**关闭按钮**，可供用户**隐藏**此警告文字框。

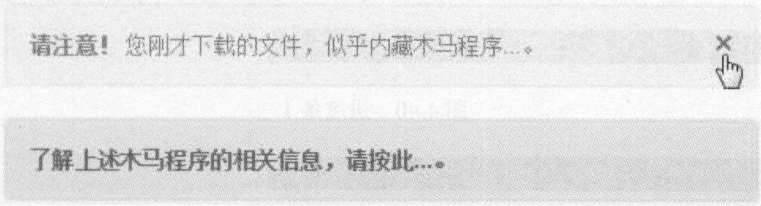


图 4-38 警告文字框

当单击警告文字框右侧的**关闭按钮**之后，可以看到第一个警告文字框消失了，如图 4-39 所示：

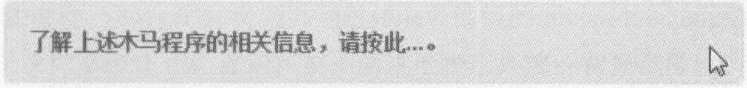


图 4-39 单击关闭按钮后警告文字框消失

```
    <div class="alert alert-warning alert-dismissible" role="alert">
      <button type="button" class="close" data-dismiss="alert" aria-label=
"Close">
      <span aria-hidden="true">&times;</span></button>
      <strong>请注意！</strong>您刚才下载的文件，似乎内藏木马程序...。
    </div>

    <div class="alert alert-success" role="alert">
      <a href="#" class="alert-link">了解上述木马程序的相关信息，请按此...。</a>
    </div>
```

在 div 元素中，设置其 class 属性值为如下组合。

- alert，表示一个警告文字框组件。
- alert-warning，可使得其警告文字框组件应用 warning 颜色样式。

- alert-dismissible，可使得其警告文字框组件的右侧额外搭建 button 元素，并分别设置其 class 属性值为 close、其 data-dismiss 属性值为 alert、其 aria-label 属性值为 Close 的【关闭按钮】，可供用户隐藏此警告文字框内容。

在 class 属性值为 alert 的 div 元素中，也可安排其 class 属性值为 alert-link 的超链接 a 元素，使得其文字外观更加搭配警告文字框的颜色样式。

4.13 进度条

本节的程序代码都来自于 Ch04-13-progress-bar.html 范例文件。

在图 4-40 中的进度条组件带有**总进度的浅灰色长条**和**表示当前进度的特定颜色样式的长条**。其中第 2 个进度条组件（见图 4-41）还带有**表示当前进度的数值百分比**。

图 4-40 进度条 1

图 4-41 进度条 2

```
<div class="progress">
   <div class="progress-bar" role="progressbar"
      aria-valuenow="60" aria-valuemin="0"
      aria-valuemax="100" style="width: 60%;">
   </div>
</div>

<div class="progress">
   <div class="progress-bar" role="progressbar"
      aria-valuenow="60" aria-valuemin="0"
      aria-valuemax="100" style="width: 60%;">
     60%
   </div>
</div>
```

通过 class 属性值分别为 progress 和 progress-bar 的前后两层 div 元素来产生进度条组件。其中，内层 div 元素还额外设置：

- role 属性值为 progressbar。
- aria-valuenow 属性值为 60，表示当前进度为 60%。
- aria-valuemin 属性值为 0，表示一开始的最低进度为 0%。
- aria-valuemax 属性值为 100，表示最后完成的进度是 100%。
- style 属性值为 width: 60%，可让进度条显示已达 60% 左右时的进度指示外观。

为了让进度条内也显示出 60%的字样，可以在 class 属性值为 progress-bar 的 div 元素的内容区中加上【60%】的文字。

在图 4-42 的进度条组件里，因为当前进度的较小背景范围不足以容纳当前进度的数值百分比，只好额外安排较大的背景范围，以容纳 0% 或 2% 字样的数值百分比。

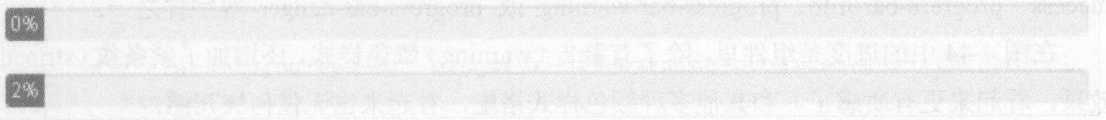


图 4-42 进度条内显示当前进度的百分比

```
<div class="progress">
  <div class="progress-bar" role="progressbar"
    aria-valuenow="0" aria-valuemin="0"
    aria-valuemax="100" style="min-width: 2em;">
    0%
  </div>
</div>

<div class="progress">
  <div class="progress-bar" role="progressbar"
    aria-valuenow="2" aria-valuemin="0"
    aria-valuemax="100" style="min-width: 2em; width: 2%;">
    2%
  </div>
</div>
```

若当前的加载进度仍然为 0%，可额外通过style属性值的min-width: 2em或min-width: 2em ; width: 2% 设置，这样至少让 0% 显示在狭小的进度指示外观的背景范围内。

图 4-43 中的进度条组件应用了不同颜色样式，分别为成功（success）和信息（info）样式。

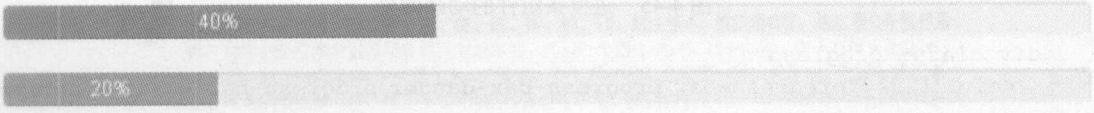


图 4-43 进度条组件应用了不同颜色样式

```
<div class="progress">
  <div class="progress-bar progress-bar-success"
    role="progressbar" aria-valuenow="40"
    aria-valuemin="0" aria-valuemax="100" style="width: 40%">
    40%
  </div>
</div>

<div class="progress">
  <div class="progress-bar progress-bar-info"
    role="progressbar" aria-valuenow="20"
```

```
        aria-valuemin="0" aria-valuemax="100" style="width: 20%">
        20%
      </div>
    </div>
```

为了改变进度条指示外观的色泽，可将其内层 div 元素的 class 属性值加上 progress-bar-success、progress-bar-info、progress-bar-warning 或 progress-bar-danger 等字样之一。

在图 4-44 中的进度条组件里，除了有警告（warning）颜色样式，还增加了**斜条纹**（striped）效果，看起来更有质感了。和其他多种颜色样式搭配，看起来也无任何突兀感。

图 4-44　进度条组件应用了斜条纹效果

```
<div class="progress">
  <div class="progress-bar progress-bar-warning progress-bar-striped"
    role="progressbar" aria-valuenow="60" aria-valuemin="0"
    aria-valuemax="100" style="width: 60%">
    60%
  </div>
</div>
```

为了让进度条上出现横条纹，可在其内层 div 元素的 class 属性值上加上 progress-bar-striped 字样。

图 4-45 中的进度条组件里，除了危险（danger）颜色样式、斜条纹（striped）效果，还加入了**处于活动中**（active）状态所显示出来的动画特效，让整个进度条组件看起来有斜条纹，也有**位移特效**。

图 4-45　进度条组件的动画效果

```
<div class="progress">
  <div class="progress-bar progress-bar-danger progress-bar-striped active"
role=
    "progressbar" aria-valuenow="80" aria-valuemin="0" aria-valuemax="100"
    style="width: 80%">
    80%
  </div>
</div>
```

为了让进度条上出现横条纹的**位移特效**，可在其内层 div 元素的 class 属性值上加注 progress-bar-striped **active**。

图 4-46 中的**组合式**进度条组件其实是由多个附属的进度条组件所构成的。各个附属的进度条组件可拥有各自的颜色样式、长度、当前进度的数值和说明文字。

图 4-46 组合式进度条组件

```
<div class="progress">
  <div class="progress-bar progress-bar-success" style="width: 35%">
    35% 下载
</div>

  <div class="progress-bar progress-bar-warning progress-bar-striped"
     style="width: 20%">
    20% 解压缩
  </div>

  <div class="progress-bar progress-bar-danger" style="width: 15%">
    15% 安装中 ...
  </div>
</div>
```

组合式的进度条可在不同区段中应用不同的色泽，并标注对应的提示文字，例如【35% 下载】、【20% 解压缩】、【15% 安装中】。

4.14 媒体对象

本节的程序代码都来自于 Ch04-14-media-object.html 范例文件。

4.14.1 一般图片列表

在图 4-47 中的一般图片列表组件里共有 3 个列表项。列表项之间略微留有间距，每个列表项中都带有**最左侧的缩略图**、**标题文字**和**文字内容**。

西红柿

西红柿含有茄红素、类胡萝卜素、磷、铁、钾、钠、镁、维生素A、维生素B群、维生素C等营养素。西红柿中的茄红素含量是所有蔬果中最高的，而且一个西红柿只有25卡路里，每天吃两个西红柿(约300克)，就可以满足体内1天维生素C的需求。西红柿中的茄红素是一种抗氧化剂，有助于推迟老化；所含的类胡萝卜素、维生素C则可以增强血管功能，预防血管老化，与叶酸均有益于维持皮肤健康。另外，西红柿的纤维质含量高，可以预防男性前列腺癌和女性乳房癌。西红柿中含有大量的果胶和柿胶酚等可溶性收敛剂，容易和胃酸发生作用而凝结，增加胃部压力，易造成胃胀痛或呕吐，空腹时最好不要食用。西红柿每天适合食用量为2~3个。

苹果

苹果为美国癌症学会推广的30种抗癌蔬果中，排名第一。以纤维和果胶功效突出，含钾量也高。依据食品营养成分数据库中记载(五爪苹果；如表一)，植物性化学物质则有胡萝卜素、泛酸(鞣酸、苹果酸、柠檬酸、硒石酸、枸橼酸、丹宁酸、苹果香醇(香气)。在红色和紫红色的苹果中含有山茶酚及槲皮素(两者存在于果皮内)、植物性凝血素(Lectin)、β-胡萝卜素、茄红素。

樱桃

樱桃营养特别丰富，果实富含糖、蛋白质、维生素及钙、铁、磷、钾等多种元素。樱桃性热，味甘，具有益气、健脾、和胃、祛风湿的功效。常食樱桃可补充体内对铁元素的需求，促进血红蛋白再生，既可防治缺铁性贫血，又可增强体质，健脑益智，还能养颜驻容，使皮肤红润嫩白，去皱消斑。

图 4-47 一般图片列表

```
<div class="media">

  <div class="media-left media-top">
    <a href="#">
      <img class="media-object" src="images/西红柿.jpg"
          alt="..." width="50" height="50">
    </a>
</div>

  <div class="media-body">
    <h4 class="media-heading">西红柿</h4>

    西红柿含有茄红素、类胡萝卜素、磷、铁、钾、钠、镁、维生素A、维生素B群、
    维生素C等营养素。西红柿中的茄红素含量是所有蔬果中最高的，而且一个西红柿
    只有25卡路里，每天吃两个西红柿(约300克)，就可以满足体内1天维生素C的需
    求。
    西红柿中的茄红素是一种抗氧化剂，有助于推迟老化；所含的类胡萝卜素、维生素C
    则可以增强血管功能，预防血管老化，与叶酸均有益于维持皮肤健康。另外，西红柿
    的纤维质含量高，可以预防男性前列腺癌和女性乳房癌。
    西红柿中含有大量的果胶和柿胶酚等可溶性收敛剂，容易和胃酸发生作用而凝结，增
    加胃部压力，易造成胃胀痛或呕吐，空腹时最好不要食用。西红柿每天适合食用量为
    2~3个。
  </div>

</div>
```

上述一般图片列表主要是由 class 属性值分别为 **media** 和 **media-left media-top** 的前后**两层**的 div 元素以及**内层** 2 个 div 元素所构成的单项组件。其中：

- 内层第一个 div 元素的 class 属性值为 media-left media-top，表示此图片列表的单项组件为左侧对齐的最上方。
- 内层第二个 div 元素的 class 属性值为 media-body，内含此单项组件的内容，其内部还可设置 class 属性值为 media-heading 的标题文字。

如下程序代码生成的图片列表的单项组件，和上述的不同点在于：内层第一个 div 元素的 class 属性值内含的不是 media-top，而是media-middle，表示此图片列表的单项组件为**左侧**对齐的**中间**。

```
<div class="media">

  <div class="media-left media-middle">
    <a href="#">
      <img class="media-object" src="images/苹果.jpg" alt="..." width="50"
height="50">
    </a>
  </div>
```

```
    <div class="media-body">
      <h4 class="media-heading">苹果</h4>

      苹果为美国癌症学会推广的 30 种抗癌蔬果中，排名第一。以纤维和果胶功效突出，
      含钾量也高。依据食品营养成分数据库中记载(五爪苹果；如表一)，植物性化学物质
      则有胡萝卜素、泛酸(鞣酸、苹果酸、柠檬酸、硒石酸、枸橼酸、丹宁酸、苹果香醇(香
      气)。在红色和紫红色的苹果中含有山茶酚及槲皮素(两者存在于果皮内)、植物性凝血
      素(Lectin)、β-胡萝卜素、茄红素。
    </div>

  </div>
```

如下程序代码生成的图片列表的单项组件，和上述的不同点在于：内层第一个 div 元素的 class 属性值内含的不是 media-top或 media-middle，而是 media-bottom，表示此图片列表的单项组件为**左侧**对齐的**底部**。

```
  <div class="media">

    <div class="media-left media-bottom">
      <a href="#">
        <img class="media-object" src="images/樱桃.jpg"
            alt="..." width="50" height="50">
      </a>
    </div>

    <div class="media-body">
      <h4 class="media-heading">樱桃</h4>

      樱桃营养特别丰富，果实富含糖、蛋白质、维生素及钙、铁、磷、钾等多种元素。樱
      桃性热，味甘，具有益气、健脾、和胃、祛风湿的功效。常食樱桃可补充体内对铁元
      素的需求，促进血红蛋白再生，既可防治缺铁性贫血，又可增强体质，健脑益智，还
      能养颜驻容，使皮肤红润嫩白，去皱消斑。
    </div>

  </div>
```

此图片列表单项组件的程序代码结构是由多个**两层**的 div 元素搭配其他元素所构成的，已经显得有些复杂，请读者在处理其相关程序代码的时候谨慎编写和修改。

4.14.2 多层次图片列表

在图 4-47 中的多层次图片列表组件里，每个列表项之间也有细微的间距，而每个列表项也含有最左侧的缩略图、标题文字和文字内容。

然而，图 4-48 中的图片列表组件却含有 **3 个层次**的列表项。

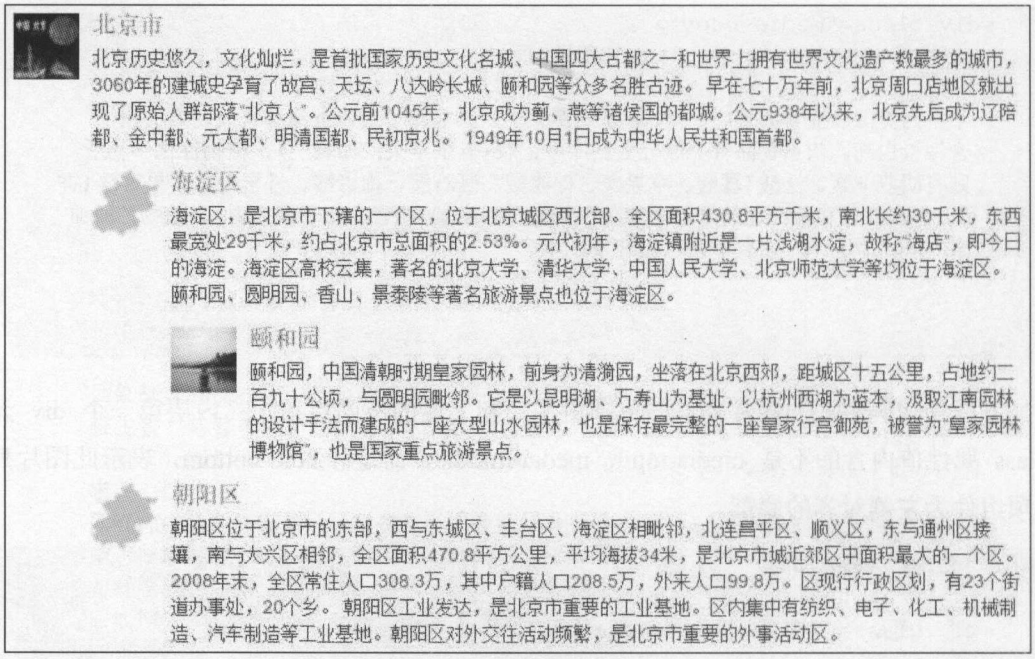


图 4-48　多层次图片列表

```
<div class="bs-example" data-example-id="media-list">
  <ul class="media-list">

    <li class="media">

      <div class="media-left">
        <a href="#">
          <img class="media-object" src="images/北京市图片.png"
              alt="..." width="50" height="50">
        </a>
      </div>

      <div class="media-body">

        <h4 class="media-heading">北京市</h4>

        <p>北京历史悠久，文化灿烂，是首批国家历史文化名城、中国四大古都之一和世界上
           拥有世界文化遗产数最多的城市，3060 年的建城史孕育了故宫、天坛、八达岭
           长城、颐和园等众多名胜古迹。 早在七十万年前，北京周口店地区就出现了原始
           人群部落“北京人”。公元前 1045 年，北京成为蓟、燕等诸侯国的都城。公元 938
           年以来，北京先后成为辽陪都、金中都、元大都、明清国都、民初京兆。 1949 年
           10 月 1 日成为中华人民共和国首都。</p>

        <!—嵌套媒体对象 -->
        <div class="media">
```

```
    <div class="media-left">
      <a href="#">
        <img class="media-object" src="images/海淀区图片.jpg"
            alt="..." width="50" height="50">
      </a>
    </div>

  <div class="media-body">

    <h4 class="media-heading">海淀区</h4>
    海淀区，是北京市下辖的一个区，位于北京城区西北部。全区面积 430.8 平方千米，
    南北长约 30 千米，东西最宽处 29 千米，约占北京市总面积的 2.53%。元代初年，
    海淀镇附近是一片浅湖水淀，故称“海店”，即今日的海淀。海淀区高校云集，著名的
    北京大学、清华大学、中国人民大学、北京师范大学等均位于海淀区。颐和园、圆明园、
    香山、景泰陵等著名旅游景点也位于海淀区。

  <!-嵌套媒体对象 -->
  <div class="media">

    <div class="media-left">
      <a href="#">
        <img class="media-object" src="images/颐和园.jpg"
            alt="..." width="50" height="50">
      </a>
    </div>

      <div class="media-body">
        <h4 class="media-heading">龙山寺</h4>
        颐和园，中国清朝时期皇家园林，前身为清漪园，坐落在北京西郊，距城区十五
        公里，占地约二百九十公顷，与圆明园毗邻。它是以昆明湖、万寿山为基址，以
        杭州西湖为蓝本，汲取江南园林的设计手法而建成的一座大型山水园林，也是保
        存最完整的一座皇家行宫御苑，被誉为“皇家园林博物馆”，也是国家重点旅游
        景点。
      </div>

    </div>

  </div><!--带有 media-body 类样式的【内层】div 元素的结尾处 -->
</div><!--带有 media 类样式的【内层】 div 元素的结尾处 -->

    <!--嵌套媒体对象 -->
    <div class="media">
      <div class="media-left">
        <a href="#">
          <img class="media-object" src="images/朝阳区图片.jpg"
              alt="..." width="50" height="50">
        </a>
      </div>
```

```
                <div class="media-body">
                  <h4 class="media-heading">朝阳区</h4>
                  朝阳区位于北京市的东部，西与东城区、丰台区、海淀区相毗邻，北连昌平区、
                  顺义区，东与通州区接壤，南与大兴区相邻，全区面积 470.8 平方公里，平均
                  海拔 34 米，是北京市城近郊区中面积最大的一个区。2008 年末，全区常住人口
                  308.3 万，其中户籍人口 208.5 万，外来人口 99.8 万。区现行行政区划，有
                  23 个街道办事处，20 个乡。 朝阳区工业发达，是北京市重要的工业基地。区内
                  集中有纺织、电子、化工、机械制造、汽车制造等工业基地。朝阳区对外交往活动
                  频繁，是北京市重要的外事活动区。
                </div>
            </div><!--带有 media 类样式的【内层】 div 元素的结尾处 -->

        </div><!--带有 media-body 类样式的【较上层】 div 元素的结尾处 -->

      </li>

    </ul>

  </div>
```

有别于一般图片列表，此处的多层次图片列表在结构上变得复杂许多，而且应用了 class 属性值为 media 的元素——不再拘泥于 div 元素，而改用了 li 元素。

此外，在一般图片列表的程序代码结构之外，还要额外加上 class 属性值为 bs-example 的 div 元素和 class 属性值为 media-list 的 ul 元素。比较复杂的结构在于：

- class 属性值为 media，会出现在前后至少两个层次的 li 元素和 div 元素上。
- class 属性值为 media-left，会出现在前后至少两个层次的 div 元素上。
- class 属性值为 media-body，会出现在前后至少两个层次的 div 元素上。

在本范例中的多层次图片列表组件里，div 元素搭配 ul 元素和其内部的 li 元素，会使得其程序代码结构变得非常复杂。请读者能更有耐心地编写和测试其程序代码。

4.15 列表分组

本节的程序代码都来自于 Ch04-15-list-group.html 范例文件。

4.15.1 默认的列表分组

在图 4-49 中的列表分组里，其实是应用了特殊样式的**单一**列表而已。在默认情况下，处于**【活动中】**状态的列表项是带有显眼的颜色样式的。

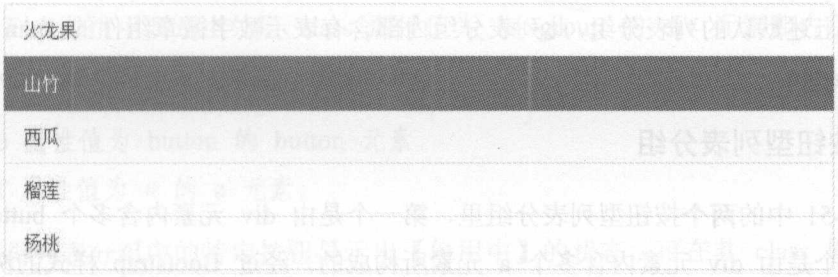


图 4-49 默认的列表分组

```
<ul class="list-group">
   <li class="list-group-item">火龙果</li>
   <li class="list-group-item active">山竹</li>
   <li class="list-group-item">西瓜</li>
   <li class="list-group-item">榴莲</li>
   <li class="list-group-item">杨桃</li>
</ul>
```

在这个范例程序中主要是通过 class 属性值为 list-group 的 ul 元素和其内部的 class 属性值为 list-group-item 的 li 元素来构成列表分组组件。要让特定 li 元素显示出【**处于活动中**】的外观，可在其 class 属性值中加设 active。

4.15.2 加上数值徽章的列表分组

在图 4-50 中的列表分组里，每个列表项还内含最右侧的**数值徽章**，以表示该项所表示的事物还剩余或已累积的数量。

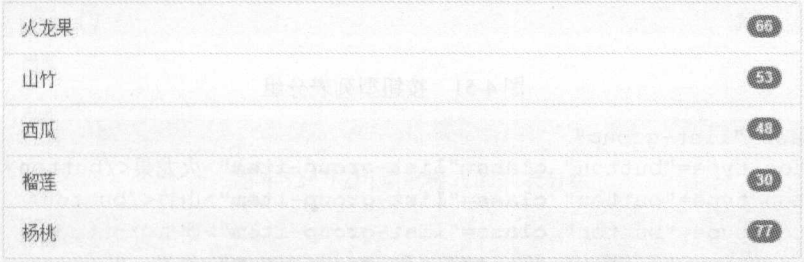


图 4-50 加上数值徽章的列表分组

```
<ul class="list-group">
   <li class="list-group-item">火龙果 <span class="badge">66</span></li>

   <li class="list-group-item">山竹 <span class="badge">53</span></li>

   <li class="list-group-item">西瓜 <span class="badge">48</span></li>

   <li class="list-group-item">榴莲 <span class="badge">30</span></li>

   <li class="list-group-item">杨桃 <span class="badge">77</span></li>
</ul>
```

有别于上述默认的列表分组，此列表分组内部含有表示数字徽章组件的 span 元素，class 属性值为 badge。

4.15.3 按钮型列表分组

在图 4-51 中的**两个**按钮型列表分组里，第一个是由 div 元素内含多个 button 元素所构成的；另一个是由 div 元素内含多个 a 元素所构成的。经过 Bootstrap 样式的洗礼，当鼠标指针移进上述的 button 元素和 a 元素的范围内时，鼠标指针会变成**手掌**形状，以暗示用户可以进行操作。

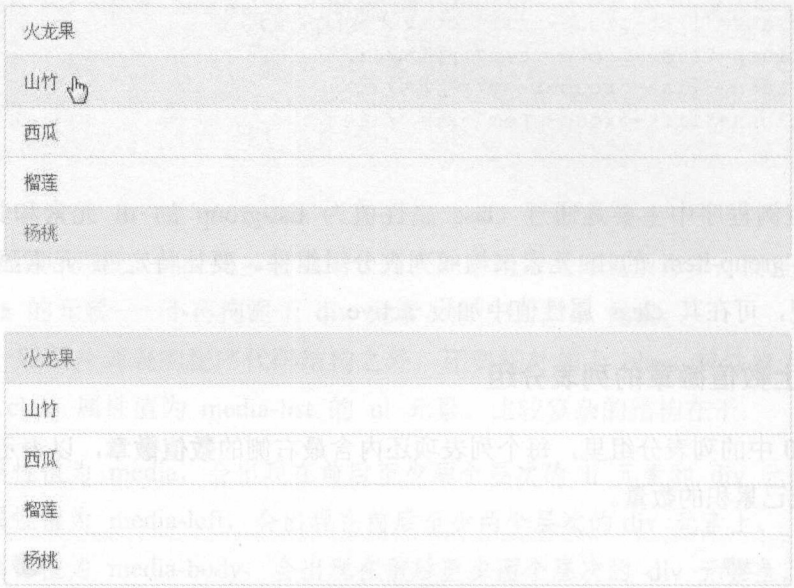


图 4-51 按钮型列表分组

```
<div class="list-group">
   <button type="button" class="list-group-item">火龙果</button>
   <button type="button" class="list-group-item">山竹</button>
   <button type="button" class="list-group-item">西瓜</button>
   <button type="button" class="list-group-item">榴莲</button>

   <button type="button" class="list-group-item disabled">杨桃</button>
</div>

<div class="list-group">
   <a href="#" class="list-group-item disabled">火龙果</a>
   <a href="#" class="list-group-item">山竹</a>
   <a href="#" class="list-group-item">西瓜</a>
   <a href="#" class="list-group-item">榴莲</a>
   <a href="#" class="list-group-item">杨桃</a>
</div>
```

按钮型列表分组主要是由外层 class 属性值为 list-group 的 div 元素和内层 class 属性值为 list-group-item 的如下元素之一所构成的。

- type 属性值为 button 的 button 元素。
- href 属性值为 # 的 a 元素。

若要使得列表分组中的特定按钮显示出【停用中】的状态，可在其 class 属性值中加注 disabled。

4.15.4 应用颜色样式的列表分组

在图 4-52 中的列表分组里，第一组是由 ul 元素内含 li 元素所构成的；第二组是由 div 元素内含 a 元素所构成的。因此，在第二组列表分组的各项上，会使得鼠标指针显示**手掌**形状。此外，如下两个列表分组的各个项上，都应用了不同的颜色样式。

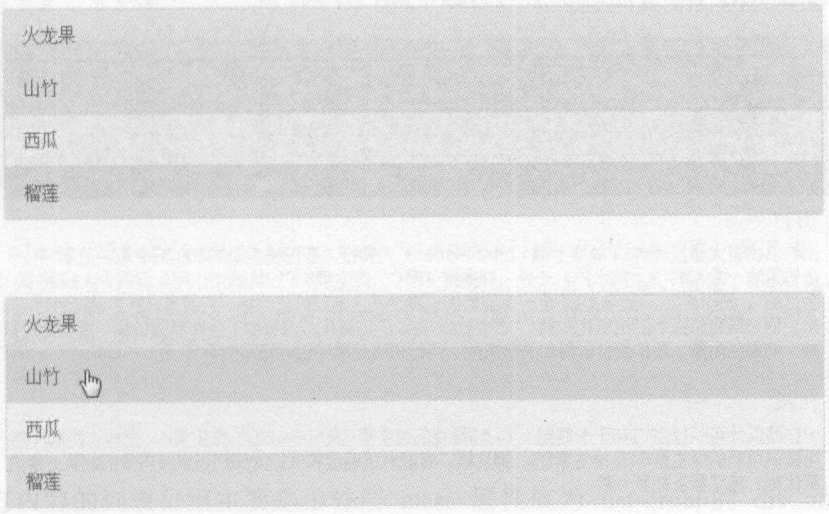


图 4-52 应用颜色样式的列表分组

```
<ul class="list-group">
   <li class="list-group-item list-group-item-success">火龙果</li>
   <li class="list-group-item list-group-item-info">山竹</li>

   <li class="list-group-item list-group-item-warning">西瓜</li>

   <li class="list-group-item list-group-item-danger">榴莲</li>
</ul>

<p style="height: 30px"></p>

<div class="list-group">
   <a href="#" class="list-group-item list-group-item-success">火龙果</a>
```

```
    <a href="#" class="list-group-item list-group-item-info">山竹</a>
    <a href="#" class="list-group-item list-group-item-warning">西瓜</a>
    <a href="#" class="list-group-item list-group-item-danger">榴莲</a>
</div>
```

上述应用颜色样式的列表分组主要是由外层 class 属性值为 list-group 的 ul 元素和内层 class 属性值为 list-group-item list-group-item-success 的 li 元素或 a 元素所构成的。其中 *list-group-item-success* 也可修改成 *list-group-item-info* 、 *list-group-item-warning* 或 list-group-item-danger，以改变其列表项的背景外观。

4.15.5 带有内容的列表分组

在图 4-53 中的列表分组里，各列表分组的各项都带有**标题**文字和**文字内容**，而**【处于活动中】**状态的选项则有与其他选项不同的**醒目颜色样式**。

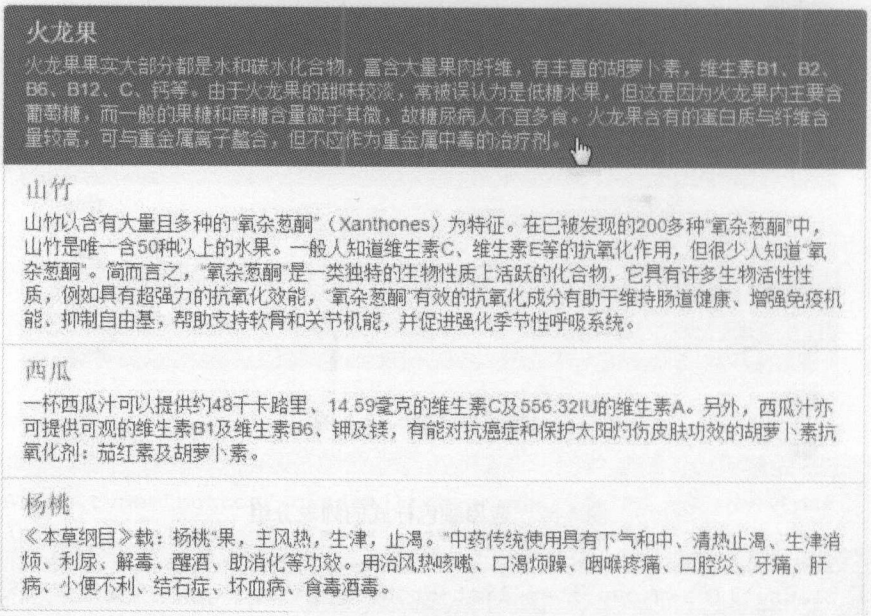


图 4-53 带有内容的列表分组

```
<div class="list-group">
  <a href="#" class="list-group-item active">
    <h4 class="list-group-item-heading">火龙果</h4>
    <p class="list-group-item-text">火龙果果实大部分都是水和碳水化合物，
      富含大量果肉纤维，有丰富的胡萝卜素，维生素 B1、B2、B6、B12、C、钙等。
      由于火龙果的甜味较淡，常被误认为是低糖水果，但这是因为火龙果内主要含葡萄
      糖，而一般的果糖和蔗糖含量微乎其微，故糖尿病人不宜多食。火龙果含有的蛋白
      质与纤维含量较高，可与重金属离子螯合，但不应作为重金属中毒的治疗剂。</p>
```

```
    </a>

    <a href="#" class="list-group-item">
      <h4 class="list-group-item-heading">山竹</h4>

      <p class="list-group-item-text">山竹以含有大量且多种的“氧杂蒽酮”(Xanthones)
        为特征。在已被发现的 200 多种“氧杂蒽酮”中，山竹是唯一含 50 种以上的水果。
        一般人知道维生素 C、维生素 E 等的抗氧化作用，但很少人知道“氧杂蒽酮”。简
        而言之，“氧杂蒽酮”是一类独特的生物性质上活跃的化合物，它具有许多生物活
        性性质，例如具有超强力的抗氧化效能，“氧杂蒽酮”有效的抗氧化成分有助于维
        持肠道健康、增强免疫机能、抑制自由基，帮助支持软骨和关节机能，并促进强化
        季节性呼吸系统。</p>
    </a>

    <a href="#" class="list-group-item">
      <h4 class="list-group-item-heading">西瓜</h4>

      <p class="list-group-item-text">一杯西瓜汁可以提供约 48 千卡路里、14.59 毫克的
        维生素 C 及 556.32IU 的维生素 A。另外，西瓜汁亦可提供可观的维生素 B1 及维生素 B6、
        钾及镁，有能对抗癌症和保护太阳灼伤皮肤功效的胡萝卜素抗氧化剂：茄红素及胡萝卜素。
        </p>
    </a>

    <a href="#" class="list-group-item">
      <h4 class="list-group-item-heading">杨桃</h4>

      <p class="list-group-item-text">《本草纲目》载：杨桃“果，主风热，生津，止渴。”
        中药传统使用具有下气和中、清热止渴、生津消烦、利尿、解毒、醒酒、助消化等
        功效。用治风热咳嗽、口渴烦躁、咽喉疼痛、口腔炎、牙痛、肝病、小便不利、结
        石症、坏血病、食毒酒毒。</p>
    </a>

</div>
```

上述带有内容的列表分组主要是由外层 class 属性值为 list-group的 div 元素和内层如下元素所构成。

- class 属性值为 list-group-item 的 a 元素。
- class 属性值为 list-group-item-heading 的 h1 至 h6 元素之一。
- class 属性值为 list-group-item-text 的 p 段落元素。

4.16 面板

本节的程序代码都来自于 Ch04-16-panels.html 范例文件。

4.16.1 一般面板

对于图 4-54 中的 3 个**面板**而言：

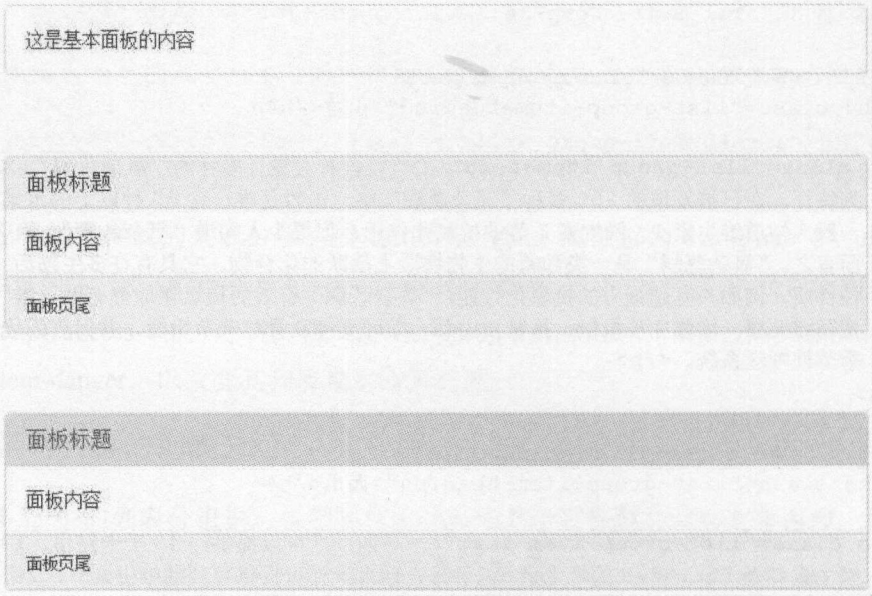


图 4-54　一般面板

- 第一个面板只有内容区。
- 第二个面板拥有标题区、内容区和页尾区。
- 第三个面板除了拥有标题区、内容区和页尾区之外，标题区还应用了危险（danger）颜色样式。

```
<div class="panel panel-default">
    <div class="panel-body">
      这是基本面板的内容
    </div>
</div>

<p style="height: 30px"></p>

<div class="panel panel-default">

    <div class="panel-heading">
      <h3 class="panel-title">面板标题</h3>
    </div>

    <div class="panel-body">
      面板内容
    </div>

    <div class="panel-footer">
      <small>面板页尾</small>
    </div>
```

```
</div>

<p style="height: 30px"></p>

<div class="panel panel-danger">

    <div class="panel-heading">
      <h3 class="panel-title">面板标题</h3>
    </div>

    <div class="panel-body">
      面板内容
    </div>

    <div class="panel-footer">
      <small>面板页尾</small>
    </div>

</div>
```

上述面板主要是由如下元素共同构成的。

- class 属性值为 panel panel-default 的 div 元素。
- class 属性值为 panel-heading 的 div 元素。
- class 属性值为 panel-title 的 h1 至 h6 元素之一。
- class 属性值为 panel-body 的 div 元素。
- class 属性值为 panel-footer 的 div 元素。

4.16.2 带有表格内容的面板

在图 4-55 中的面板里，除了拥有**标题区**之外，其**内容区**还内含一个 **4 行 4 列**的**表格**结构，其中第一行是标题栏，其内部每个单元格的文字内容都为**粗体**字。

面板标题

编号	表格标题 1	表格标题 2	表格标题 3
1	...	...	...
2	...	...	...
3	...	...	...

图 4-55 带有表格内容的面板

```
<div class="panel panel-default">

   <div class="panel-heading">面板标题</div>

   <table class="table">

     <thead>
       <tr>
         <th>编号</th>
         <th>表格标题 1</th>
         <th>表格标题 2</th>
         <th>表格标题 3</th>
       </tr>
     </thead>

     <tbody>
       <tr>
         <th scope="row">1</th>
         <td>...</td>
         <td>...</td>
         <td>...</td>
       </tr>

       <tr>
         <th scope="row">2</th>
         <td>...</td>
         <td>...</td>
         <td>...</td>
       </tr>

       <tr>
         <th scope="row">3</th>
         <td>...</td>
         <td>...</td>
         <td>...</td>
         </tr>
         </tbody>
   </table>

</div>
```

上述带有表格内容的面板主要是由如下元素共同构成的。

- class 属性值为 panel panel-default 的 div 元素。
- class 属性值为 panel-heading 的 div 元素。
- class 属性值为 table 的 table 元素。

4.16.3 带有列表分组内容的面板

在图 4-56 中的面板里，除了拥有**标题区**之外，还拥有内含**列表**组件的**内容区**。在此，列表是由 ul 元素和其内部的 li 元素所构成的。

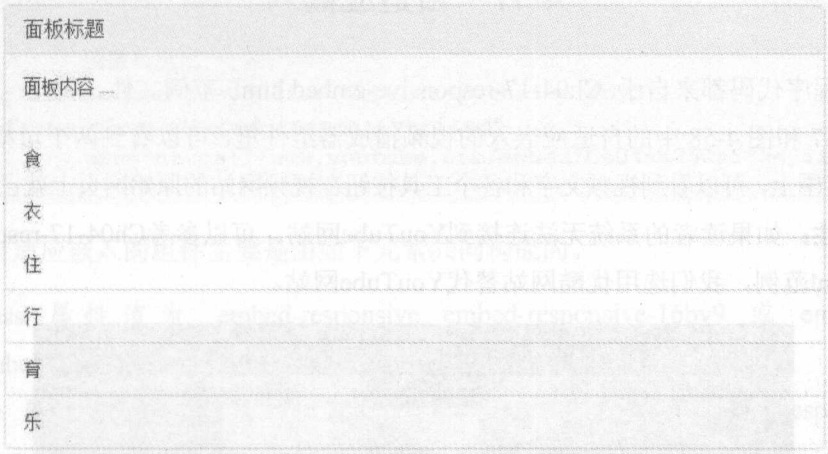


图 4-56 带有列表分组内容的面板

```
<div class="panel panel-default">

    <div class="panel-heading">
        <h3 class="panel-title">面板标题</h3>
    </div>

    <div class="panel-body">
        <p>面板内容 ...</p>
    </div>

    <ul class="list-group">
        <li class="list-group-item">食</li>
        <li class="list-group-item">衣</li>
        <li class="list-group-item">住</li>
        <li class="list-group-item">行</li>
        <li class="list-group-item">育</li>
        <li class="list-group-item">乐</li>
        </ul>

</div>
```

上述带有列表分组内容的面板主要是由如下元素共同构成的。

- class 属性值为 panel panel-default 的 div 元素。
- class 属性值为 panel-heading 的 div 元素。
- class 属性值为 panel-title 的 h1 至 h6 元素之一。

- class 属性值为 panel-body 的 div 元素。
- class 属性值为 list-group 的 ul 元素。
- class 属性值为 list-group-item 的 li 元素。

4.17 自适应嵌入

本节的程序代码都来自于 Ch04-17-responsive-embed.html 范例文件。

在图 4-57 和图 4-58 中的自适应嵌入的视频播放器组件里，可以看到两个可播放的视频组件。在视频范围上，可以看到视频文字和各个工具按钮在视频网站的原始网页上就已经设计好了。

改编者注：如果读者的系统无法连接到YouTube网站，可以参考Ch04-17-responsive-embed01.html范例，我们选用优酷网站替代YouTube网站。

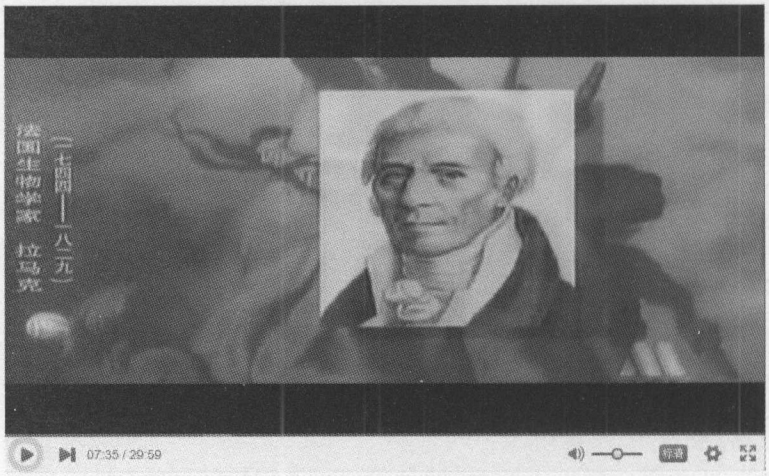


图 4-57 自适应嵌入视频播放组件 1

图 4-58 自适应嵌入视频播放组件 2

```
<!-- 16 : 9 图像长宽比 -->
<div class="embed-responsive embed-responsive-16by9">
    <iframe class="embed-responsive-item"
            src="https://www.youtube.com/embed/Ug3HBGpdeSA"></iframe>
</div>

<p style="height: 30px"></p>

<!-- 4 : 3 图像长宽比 -->
<div class="embed-responsive embed-responsive-4by3">
    <iframe class="embed-responsive-item"
            src="https://www.youtube.com/embed/CeU4aE35OaY"></iframe>
</div>
```

上述自适应嵌入的组件主要是由如下元素共同构成的。

- class 属性值为 embed-responsive embed-responsive-16by9 或 embed-responsive embed-responsive-4by3 的 div 元素。
- src 属性值为一个正确的视频网址、其 class 属性值为 embed-responsive-item 的 iframe 元素。

本范例使用 iframe 元素嵌入了视频网站上内含特定视频的网页到本范例所对应的网页程序里，这样便可产生在网页当中内含其他子网页的效果。

改编者注：如果读者的系统无法连接到YouTube网站，将上面程序代码中的视频地址用如下两个视频网址替代掉。

```
http://v.youku.com/v_show/id_XNzA0ODYyMTY=.html?f=5517394
http://v.youku.com/v_show/id_XNzM4NzM4MTY=.html?f=5517394
```

第 5 章　Zurb's Foundation 的学习与使用

在 Zurb's Foundation 的官网（http://foundation.zurb.com/）上，Zurb 公司声称自己的 Foundation 是现今于全世界中最先进的自适应网页技术的**前端 / 客户端**框架（framework），不仅可使得设计人员能更快学会和开始编写程序代码，还可让各个设备上的浏览器通过本设备上的 GPU 加速机制来加速显示网页的全貌。

5.1　Zurb's Foundation 简易应用方式

在其官网的附属页面 http://foundation.zurb.com/develop/download.html 中，可下载完整的或由程序设计人员选择下载自定义型的 Zurb's Foundation。

（1）建议下载后在本地使用。可到 http://foundation.zurb.com/docs/css.html 单击“Download Foundation CSS”按钮。

（2）进入 http://foundation.zurb.com/develop/download.html 页面后，再单击“Download Everything”按钮，即可下载一个类似 foundation-5.5.2.zip 的压缩文件。

（3）将上述 zip 文件解压缩之后，会得到名称类似 foundation-5.5.2 的文件夹，其中包含 css、img 和 js 共 3 个子文件夹。

（4）日后要应用 Zurb's Foundation 框架的网页，直接添加上述名称到类似 foundation-5.5.2 的文件夹中，并包含如下的程序代码即可：

```
…
<head>
   <meta charset="utf-8" />
   <meta name="viewport" content="width=device-width, initial-scale=1.0" />
   <title>Foundation | Welcome</title>
   <link rel="stylesheet" href="css/foundation.css" />
   <script src="js/vendor/modernizr.js"></script>
</head>
…
<body>
…
   <script src="js/vendor/jquery.js"></script>
   <script src="js/foundation.min.js"></script>
```

```
    <script>
        $(document).foundation();
    </script>
</body>
```

读者也可考虑将 <head> 和 </head> 语句拿掉，因为使用 HTML5 是可以省略 head 语句的。

5.2 滑入菜单

本节的程序代码都来自于 Ch05-02-off-canvas-menus.html 范例文件。

5.2.1 默认的滑入菜单

在图 5-1 中的滑入菜单组件里，单击左上角的【常见的类胡萝卜素】标题超链接，即可使得菜单从左往右滑入屏幕画面当中。

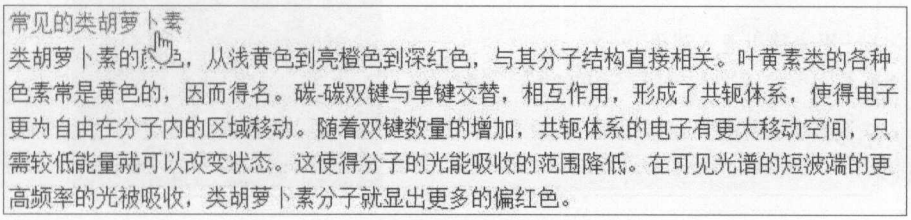


图 5-1 默认的滑入菜单

单击图 5-1 中的标题超链接之后，其画面显示如图 5-2 所示。

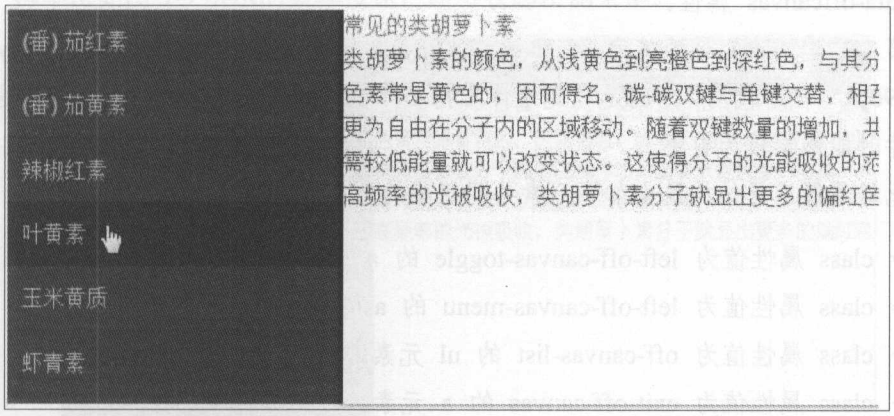


图 5-2 单击标题超链接后菜单从左往右滑入

```
<div class="off-canvas-wrap" data-offcanvas>
  <div class="inner-wrap" style="height: 290px">

    <a class="left-off-canvas-toggle" href="#">常见的类胡萝卜素</a>
```

```
    <!--滑入菜单类 -->
    <aside class="left-off-canvas-menu">
      <ul class="off-canvas-list">
        <li><a href="#">(番) 茄红素</a></li>
        <li><a href="#">(番) 茄黄素</a></li>
        <li><a href="#">辣椒红素</a></li>
        <li><a href="#">叶黄素</a></li>
        <li><a href="#">玉米黄质</a></li>
        <li><a href="#">虾青素</a></li>
      </ul>
    </aside>

    <!--主内容 -->
    <p> 类胡萝卜素的颜色，从浅黄色到亮橙色到深红色，与其分子结构直接相关。叶
        黄素类的各种色素常是黄色的，因而得名。碳-碳双键与单键交替，相互作用，
        形成了共轭体系，使得电子更为自由在分子内的区域移动。随着双键数量的增加，
        共轭体系的电子有更大移动空间，只需较低能量就可以改变状态。这使得分子的
        光能吸收的范围降低。在可见光谱的短波端的更高频率的光被吸收，类胡萝卜素
        分子就显出更多的偏红色。</p>

    <!--用来移出滑入菜单 -->
    <a class="exit-off-canvas"></a>

  </div>
</div>
```

此滑入菜单组件由如下元素共同组成。

- 最上层是 div 元素，其 class 属性值为 off-canvas-wrap，还有不带属性值的 data-offcanvas 属性。
- 第二层是 class 属性值为 inner-wrap 的 div 元素。笔者在此额外编写了 style="height: 290px" 程序代码，为的是让此组件维持 290 像素的高度，以便读者在测试这个范例文件时能感觉较为自然。
- 此组件第三层结构是由如下元素共同组成的。
  - class 属性值为 left-off-canvas-toggle 的 a 元素。
  - class 属性值为 left-off-canvas-menu 的 aside 元素。
  - class 属性值为 off-canvas-list 的 ul 元素。
  - class 属性值为 exit-off-canvas 的 a 元素。

5.2.2 双边滑入菜单

在图 5-3 中可以看到，双边滑入菜单组件的**左上角**和**右上角**都有一个**菜单按钮**。

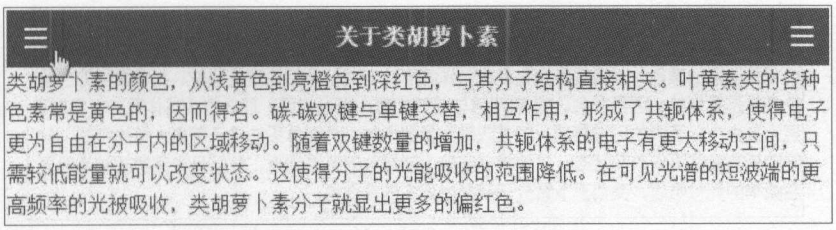


图 5-3　双边滑入菜单

通过单击**左上角**的**菜单按钮**，不仅此组件整体往右边滑动，还可看到如图 5-4 所示的**左侧**菜单。

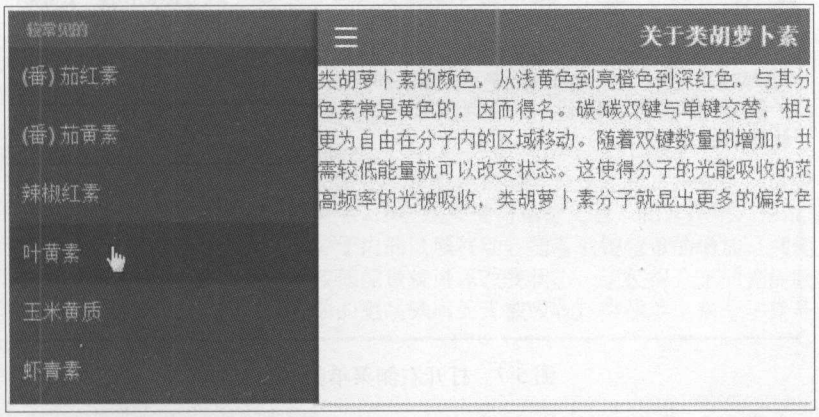


图 5-4　双边滑入菜单的左侧菜单

从图 5-4 可以看到，**左侧**菜单的每个选项都是可选择的，从鼠标指针变成【手掌】形状便可知道。再单击如图 5-5 所示的**菜单按钮**，即可收起**左侧**菜单。

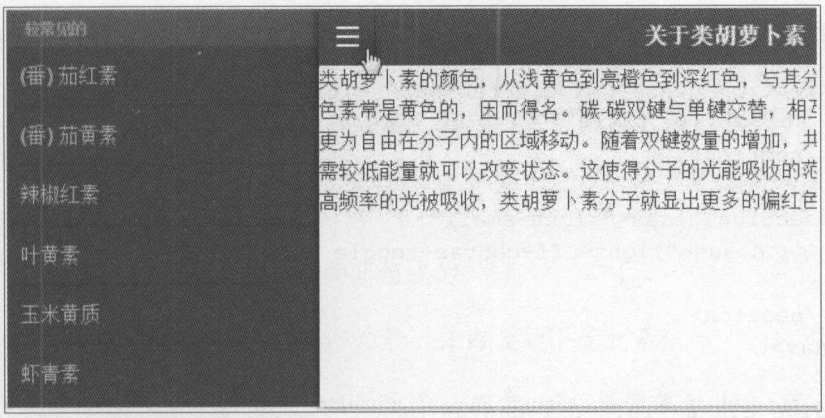


图 5-5　单击菜单按钮可以关闭滑入的左侧菜单

再单击**右上角**的**菜单按钮**，如图 5-6 所示，可打开**右侧**的菜单。

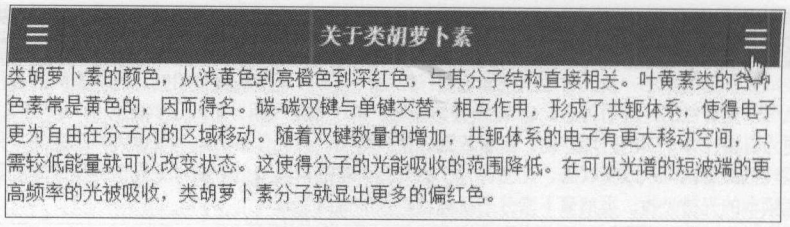


图 5-6　单击右侧的菜单按钮可以打开右侧的菜单

当**右侧**菜单被打开后，可看到如图 5-7 所示的全貌。

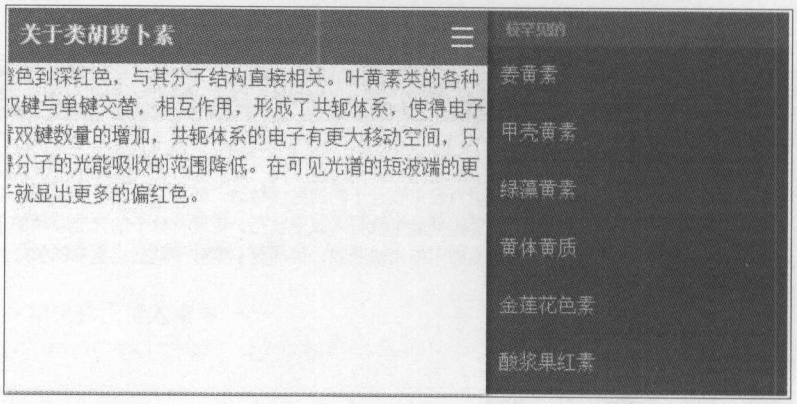


图 5-7　打开右侧菜单的情况

```
    <div class="off-canvas-wrap" data-offcanvas>
      <div class="inner-wrap" style="height: 315px">
        <nav class="tab-bar">
          <section class="left-small">
            <a class="left-off-canvas-toggle menu-icon" href="#"><span></span>
</a>
          </section>

          <section class="middle tab-bar-section">
            <h1 class="title">关于类胡萝卜素</h1>
          </section>

          <section class="right-small">
            <a class="right-off-canvas-toggle menu-icon" href="#"><span></span>
</a>
          </section>
        </nav>

        <aside class="left-off-canvas-menu">
          <ul class="off-canvas-list">
            <li><label>较常见的</label></li>
            <li><a href="#">(番) 茄红素</a></li>
            <li><a href="#">(番) 茄黄素</a></li>
            <li><a href="#">辣椒红素</a></li>
            <li><a href="#">叶黄素</a></li>
```

```
        <li><a href="#">玉米黄质</a></li>
        <li><a href="#">虾青素</a></li>
      </ul>
    </aside>

    <aside class="right-off-canvas-menu">
      <ul class="off-canvas-list">
        <li><label>较罕见的</label></li>
        <li><a href="#">姜黄素</a></li>
        <li><a href="#">甲壳黄素</a></li>
        <li><a href="#">绿藻黄素</a></li>
        <li><a href="#">黄体黄质</a></li>
        <li><a href="#">金莲花色素</a></li>
        <li><a href="#">酸浆果红素</a></li>
      </ul>
    </aside>

    <section class="main-section" style="height: 270px">
      类胡萝卜素的颜色，从浅黄色到亮橙色到深红色，与其分子结构直接相关。叶黄素
      类的各种色素常是黄色的，因而得名。碳-碳双键与单键交替，相互作用，形成了
      共轭体系，使得电子更为自由在分子内的区域移动。随着双键数量的增加，共轭体
      系的电子有更大移动空间，只需较低能量就可以改变状态。这使得分子的光能吸收
      的范围降低。在可见光谱的短波端的更高频率的光被吸收，类胡萝卜素分子就显出
      更多的偏红色。
    </section>

    <a class="exit-off-canvas"></a>
  </div>
</div>
```

比起上述默认的滑入菜单组件，本小节的**双边**滑入菜单由如下更复杂的元素所共同组成。

- 最上层是 div 元素，其 class 属性值为 off-canvas-wrap，还有不带属性值的 data-offcanvas 属性。
- 第二层是 class 属性值为 inner-wrap 的 div 元素。笔者在此额外编写了 style="height: 315px" 程序代码，为的是让此组件维持 315 像素的高度，使读者在测试这个范例文件时能感觉较为自然。
- 此组件第三层结构是由如下元素共同组成的。
  - ➢ class 属性值为 tab-bar 的 nav 元素，并内含如下子元素：
    - ◊ class属性值为left-small的section元素，内含class 属性值为 left-off-canvas-toggle menu-icon 的 a 元素。
    - ◊ class属性值为middle tab-bar-section的section元素，内含class属性值为title的 h1 至 h6 元素之一。

◊ class属性值为right-small的section元素，内含class属性值为 right-off-canvas-toggle menu-icon 的 a 元素。

- class 属性值为 left-off-canvas-menu 的 aside 元素，内含 class 属性值为 off-canvas-list 的 ul 元素。
- class 属性值为 right-off-canvas-menu 的 aside 元素，内含 class 属性值为 off-canvas-list 的 ul 元素。
- class 属性值为 main-section 的 section 元素。
- class 属性值为 exit-off-canvas 的 a 元素。

5.2.3 双边多层次滑入菜单

在图 5-8 中，可以看到此多层次滑入菜单组件的**左侧**菜单的选项 1 有下一层子菜单的提示小图标“>>”。

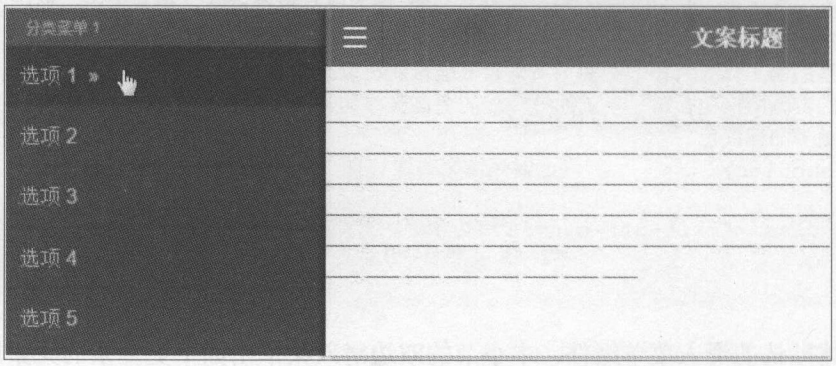


图 5-8 菜单选项 1 提示有下一层子菜单

在图 5-8 中，选择【选项 1】，可跳到选项 1 所对应的子菜单。在右侧有另一个【主】菜单，如图 5-9 所示。

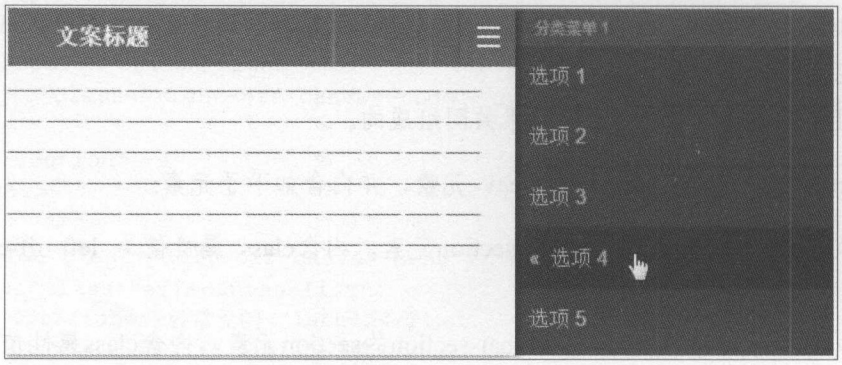


图 5-9 右侧菜单展开的情况

在图 5-9 中，选择【选项 4】，可显示如图 5-10 所示的画面。

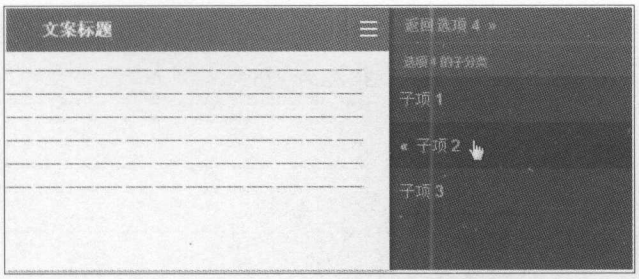


图 5-10 选项 4 的子菜单

```
<div class="off-canvas-wrap" data-offcanvas>
  <div class="inner-wrap" style="height: 270px">
    <nav class="tab-bar">
      <section class="left-small">
        <a class="left-off-canvas-toggle menu-icon"><span></span></a>
      </section>

      <section class="middle tab-bar-section">
        <h1 class="title">文案标题</h1>
      </section>

      <section class="right-small">
        <a class="right-off-canvas-toggle menu-icon"><span></span></a>
      </section>
    </nav>

    <aside class="left-off-canvas-menu">
      <ul class="off-canvas-list">
        <li><label>分类菜单 1</label></li>

        <li class="has-submenu">
          <a href="#">选项 1</a>

          <ul class="left-submenu">
            <li class="back"><a href="#">返回选项 1</a></li>

            <li><label>选项 1 的子分类</label></li>
            <li><a href="#">子项 1</a></li>

            <li class="has-submenu">
              <a href="#">子项 2</a>

              <ul class="left-submenu">
                <li class="back">
                  <a href="#">返回子项 2</a></li>

                <li><label>子项 2 的子分类</label></li>
                <li><a href="#">细项 1</a></li>
                <li><a href="#">细项 2</a></li>
              </ul>
```

```
            </li>

            <li><a href="#">子项 3</a></li>
          </ul>
        </li>

        <li><a href="#">选项 2</a></li>
        <li><a href="#">选项 3</a></li>
        <li><a href="#">选项 4</a></li>
        <li><a href="#">选项 5</a></li>
      </ul>
    </aside>

    <aside class="right-off-canvas-menu">
      <ul class="off-canvas-list">
        <li><label>分类菜单 1</label></li>

        <li><a href="#">选项 1</a></li>
        <li><a href="#">选项 2</a></li>
        <li><a href="#">选项 3</a></li>

        <li class="has-submenu">
          <a href="#">选项 4</a>

          <ul class="right-submenu">
            <li class="back"><a href="#">返回选项 4</a></li>

            <li><label>选项 4 的子分类</label></li>
            <li><a href="#">子项 1</a></li>

            <li class="has-submenu">
            <a href="#">子项 2</a>

            <ul class="right-submenu">
              <li class="back"><a href="#">返回子项 2</a></li>
              <li><label>子项 2 的子分类</label></li>
              <li><a href="#">细项 1</a></li>
              <li><a href="#">细项 2</a></li>
            </ul>
          </li>

          <li><a href="#">子项 3</a></li>
        </ul>
      </li>

      <li><a href="#">选项 5</a></li>
    </ul>
  </aside>

  <section class="main-section" style="height: 270px">
    ——————————————————————————————————————————————————————————
```

```
    ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___
    ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___
    ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___
    ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___
    ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___
    ___ ___ ___ ___ ___ ___ ___ ___
   </section>

   <a class="exit-off-canvas"></a>
  </div>
</div>
```

比起上述双边的滑入菜单组件，本小节的滑入菜单是【多层次】的，并由如下更复杂的元素所共同组成。

- 最上层是 div 元素，其 class 属性值为 off-canvas-wrap，还有不带属性值的 data-offcanvas 属性。
- 第二层是 class 属性值为 inner-wrap 的 div 元素。笔者在此额外编写了 style="height: 270px" 程序代码，是为了让此组件维持 270 像素的高度，便于读者在测试这个范例文件时能感觉较为自然。
- 此组件第三层结构是由如下元素共同组成的。

  ➢ class 属性值为 tab-bar 的 nav 元素，并内含如下子元素:

  ◊ class 属性值为 left-small 的 section 元素，内含 class 属性值为 left-off-canvas-toggle menu-icon 的 a 元素。
  ◊ class 属性值为 middle tab-bar-section 的 section 元素，内含 class 属性值为 title 的 h1 至 h6 元素之一。
  ◊ class 属性值为 right-small 的 section 元素，内含 class 属性值为 right-off-canvas-toggle menu-icon 的 a 元素。

  ➢ class 属性值为 left-off-canvas-menu 的 aside 元素，内含 class 属性值为 off-canvas-list 的 ul 元素。此 ul 元素还内含:

  ◊ class 属性值为 has-submenu 的多层次的 li 元素。
  ◊ class 属性值为 left-submenu 的多层次的 ul 元素。

  ➢ class 属性值为 right-off-canvas-menu 的 aside 元素，内含 class 属性值为 off-canvas-list 的 ul 元素。

◊ class 属性值为 has-submenu 的多层次的 li 元素。
◊ class 属性值为 right-submenu 的多层次的 ul 元素。

➢ class 属性值为 main-section 的 section 元素。
➢ class 属性值为 exit-off-canvas 的 a 元素。

5.3 置顶导航栏

本节的程序代码都来自于 Ch05-03-top-bar.html 范例文件。

在图 5-11 所示的置顶导航栏里有如下各部分。

- 导航栏名称（我的网站）。
- 导航按钮（靠左的导航按钮、靠右的导航按钮）。
- 一般按钮（自成一格的按钮、搜索）。
- 下拉式菜单按钮。
- 文本栏（关键字）。

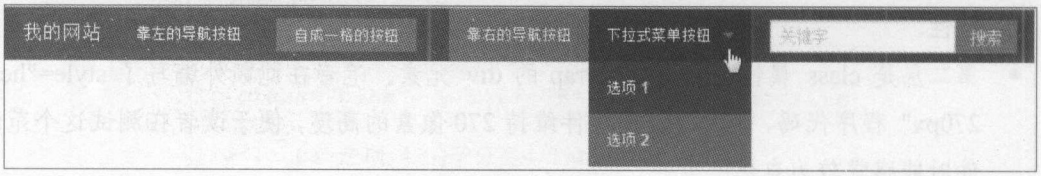


图 5-11 置顶导航栏

当浏览器的窗口宽度不够时，整个置顶导航栏会自动变成如图 5-12 所示的模样。

图 5-12 浏览器的窗口宽度不够时置顶导航栏显示的样子

单击图 5-12 右上角的展开按钮（功能按钮），会看到置顶导航栏内的各个组件变成如图 5-13 所示的**纵向**排列。

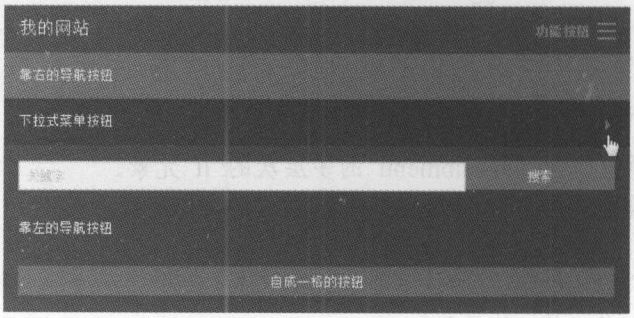


图 5-13 置顶导航展开后的样子

再单击图 5-13 中的下拉式菜单按钮，会看到菜单被打开，如图 5-14 所示。

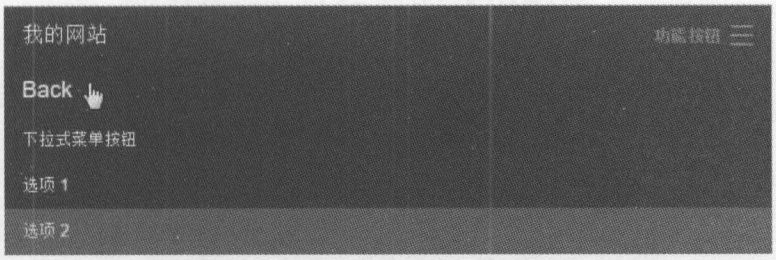


图 5-14　打开下拉式菜单

再选择图 5-14 中的 Back 选项，又回到图 5-15 所示的**纵向**置顶导航栏的模样。

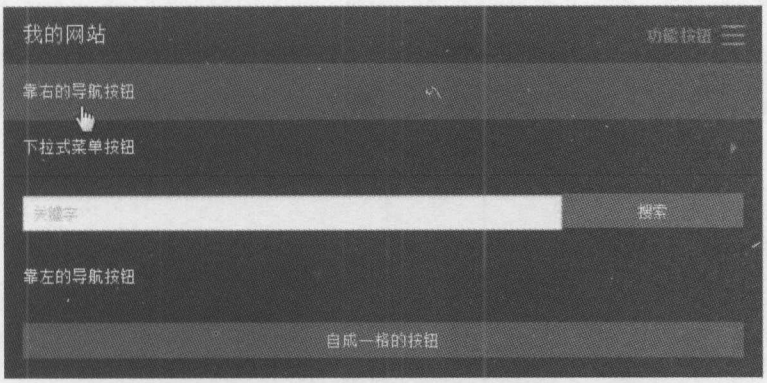


图 5-15　回到纵向置顶导航栏的模样

```
    <nav class="top-bar" data-topbar role="navigation" data-options="sticky_on:
large">

      <ul class="title-area">
        <li class="name">
          <h1><a href="#">我的网站</a></h1>
        </li>

        <li class="toggle-topbar menu-icon"><a href="#"><span>功能按钮</span></a>
</li>
      </ul>

      <section class="top-bar-section">

        <!--右侧导航区 -->
        <ul class="right">
          <li class="active"><a href="#">靠右的导航按钮</a></li>

          <li class="has-dropdown">
            <a href="#">下拉式按钮菜单</a>

            <ul class="dropdown">
```

```
            <li><a href="#">选项 1</a></li>
            <li class="active"><a href="#">选项 2</a></li>
          </ul>
        </li>

        <li class="divider"></li>

        <li class="has-form">
          <div class="row collapse">
            <div class="large-8 small-9 columns">
              <input type="text" placeholder="关键字">
            </div>

            <div class="large-4 small-3 columns">
              <a href="#" class="alert button expand">搜索</a>
            </div>
          </div>
        </li>
      </ul>

      <!--左侧导航区 -->
      <ul class="left">
        <li><a href="#">靠左的导航按钮</a></li>

        <li class="has-form">
          <a href="#" class="button">自成一格的按钮</a>
        </li>
      </ul>

    </section>
</nav>
```

本范例中的置顶导航栏主要由如下结构所共同构成。

- 第一层为 nav 元素，其 class 属性值为 top-bar，其 role 属性值为 navigation，其 data-options 属性值为 sticky_on: large，还有不带任何属性值的 data-topbar 属性。
- 第二层为如下结构。
  - ➢ class 属性值为 title-area 的 ul 元素，其内含 class 属性值分别为 name 和 toggle-topbar menu-icon 的 li 元素。
  - ➢ class 属性值为 top-bar-section 的 section 元素，其内主要由如下两个子元素所构成。
    - ◊ class 属性值为 right 的 ul 元素，内含导航按钮、下拉式按钮菜单、分隔线、搜索文本栏等。
    - ◊ class 属性值为 left 的 ul 元素，内含 class 属性值为 has-form 的 li 元素，此 li 元素又内含 class 属性值为 button 的超链接按钮。

5.4 工具栏与内置小图标

本节的程序代码都来自于 Ch05-04-icon-bars.html 范例文件。

5.4.1 跨窗口宽度的工具栏

在图 5-16 中的跨窗口宽度的工具栏里，其内部的每个工具按钮：

- 具有相同宽度而均分整个窗口宽度。
- 带有不同的小图标和提示文字。

图 5-16 跨窗口宽度的工具栏

```
<div class="icon-bar five-up">
  <a class="item">
    <img src="img/fi-home.svg">
    <label>Home</label>
  </a>

  <a class="item">
    <img src="img/fi-bookmark.svg">
    <label>Bookmark</label>
  </a>

  <a class="item">
    <img src="img/fi-info.svg">
    <label>Info</label>
  </a>

  <a class="item">
    <img src="img/fi-mail.svg">
    <label>Mail</label>
  </a>

  <a class="item">
    <img src="img/fi-like.svg">
    <label>Like</label>
  </a>

</div>
```

上述跨窗口宽度的工具栏主要是由以下结构所构成的。

- 第一层是 class 属性值为 icon-bar five-up 的 div 元素。

- 第二层是几个 class 属性值为 item 的 a 元素，并内含 src 属性值为特定图片文件路径的 img 元素以及 label 元素。

在本范例中的每个工具按钮里的小图标，引用的是本范例文件所在文件夹中的 img 子文件夹里的各个 .svg 图片文件。

5.4.2 纵向工具栏

在图 5-17 所示的纵向的工具栏里，每个工具按钮的**宽度**和**高度并不会**受到窗口宽度或高度的影响，因此它们的宽度和高度是**相同**而**固定不变**的。

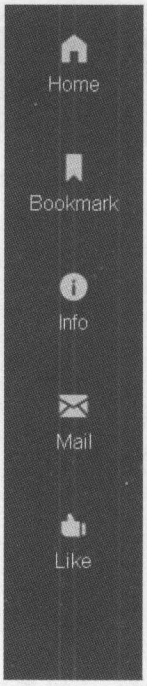


图 5-17 纵向工具栏

```
<div class="icon-bar five-up vertical">

  <a class="item">
    <img src="img/fi-home.svg">
    <label>Home</label>
  </a>

  <a class="item">
    <img src="img/fi-bookmark.svg">
    <label>Bookmark</label>
  </a>

  <a class="item">
    <img src="img/fi-info.svg">
```

```
    <label>Info</label>
  </a>

  <a class="item">
    <img src="img/fi-mail.svg">
    <label>Mail</label>
  </a>

  <a class="item">
    <img src="img/fi-like.svg">
    <label>Like</label>
  </a>

</div>
```

上述纵向工具栏主要由以下结构所构成。

- 第一层是 class 属性值为 icon-bar five-up vertical 的 div 元素。
- 第二层是几个 class 属性值为 item 的 a 元素，并内含 src 属性值为特定图片文件路径的 img 元素以及 label 元素。

5.4.3 内置图标工具按钮所构成的工具栏

在图 5-18 所示的工具栏里，各个工具按钮虽然和前两个小节对应的工具按钮在外观上看起来并没有什么差别，然而其图标其实内置于 Zurb's Foundation 的模块当中。

图 5-18 内置图标工具按钮所构成的工具栏

```
<div class="icon-bar five-up">

  <a class="item">
    <i class="fi-home"></i>
    <label>Home</label>
  </a>

  <a class="item">
    <i class="fi-bookmark"></i>
    <label>Bookmark</label>
  </a>

  <a class="item">
    <i class="fi-info"></i>
    <label>Info</label>
  </a>
```

```
    <a class="item">
      <i class="fi-mail"></i>
      <label>Mail</label>
    </a>

    <a class="item">
      <i class="fi-like"></i>
      <label>Like</label>
    </a>

</div>
```

上述内置图标工具按钮所构成的工具栏主要是由以下结构所构成的。

- 第一层是 class 属性值为 icon-bar five-up 的 div 元素。
- 第二层是几个 class 属性值为 item 的 a 元素，并内含 class 属性值为 fi-home、fi-bookmark、fi-info、fi-mail 或 fi-like 的 i 元素以及 label 元素。

在本范例中，显示各个工具按钮的小图标是浏览器通过 CSS 语句判断出 Zurb's Foundation 对于 class 属性值为 fi-home、fi-bookmark、fi-info、fi-mail 或 fi-like 所对应的图片文件**设置路径**，进而通过网络动态获取设置路径上的**图片文件**。

5.5 侧边导航栏

本节的程序代码都来自于 Ch05-05-side-nav.html 范例文件。

在图 5-19 中的侧边导航栏，看起来并不精美，但其功能性却是良好的。读者若对于 CSS 语句有所熟悉，而且具有美工底子，便可尝试去修改它的质感。

图 5-19 中的侧边导航栏具有 4 个选项，前两个选项和后两个选项之间还有一条**分隔线**。

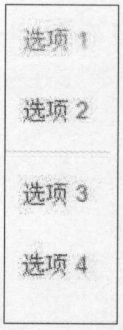


图 5-19 侧边导航栏

```
<ul class="side-nav left">

  <li class="active"><a href="#">选项 1</a></li>
  <li><a href="#">选项 2</a></li>
```

```
    <li class="divider"></li>

    <li><a href="#">选项 3</a></li>
    <li><a href="#">选项 4</a></li>

</ul>
```

上述侧边导航栏主要由如下结构所共同构成。

- 第一层是 class 属性值为 side-nav left 的 ul 元素。
- 第二层是 li 元素，必要时可设置其 class 属性值为 active，使得其具有【处于活动中】状态的外观。

在本节的范例中，其实无论是一般选项还是**分隔线**，都是由 li 元素所搭建而成的，只不过分隔线是由 class 属性值为 divider 的 li 元素所构成的。

5.6 动态置顶追踪型导航栏

本节的程序代码都来自于 Ch05-06-magellan-sticky-nav.html 范例文件。

在图 5-20 所示的动态置顶追踪型导航栏中，主要是指页面最上面的【高雄市】与【台北市】超链接所组成的这种结构。

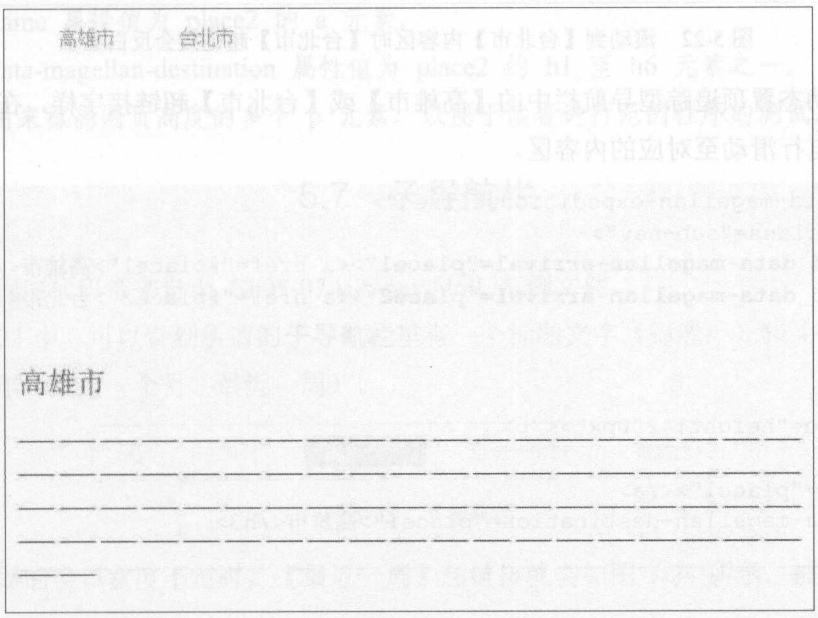


图 5-20 动态置顶追踪型导航栏

此网页被滚动至【高雄市】内容区时，动态置顶追踪型导航栏中的【高雄市】超链接字样就会突然被**反白**，如图 5-21 所示。

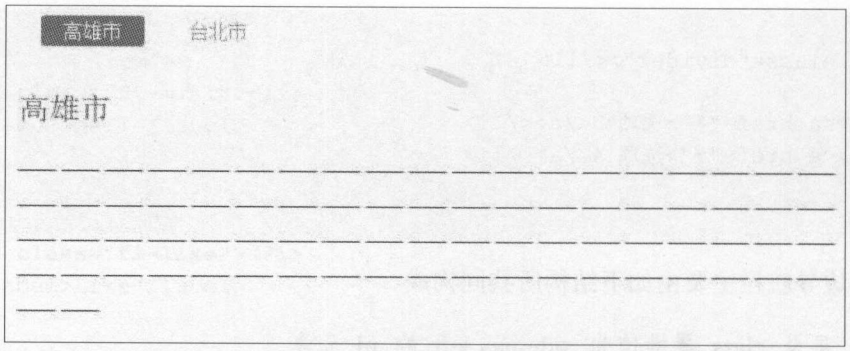


图 5-21　滚动到【高雄市】内容区时【高雄市】超链接会反白显示

此网页被滚动至【台北市】内容区时，动态置顶追踪型导航栏中的【台北市】超链接字样也会突然被反白，如图 5-22 所示。

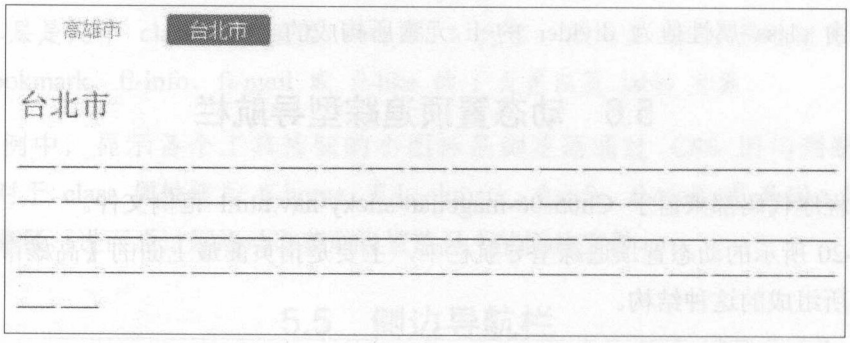


图 5-22　滚动到【台北市】内容区时【台北市】超链接会反白显示

上述的动态置顶追踪型导航栏中的【高雄市】或【台北市】超链接字样，在被选择之后，可让浏览器自行滑动至对应的内容区。

```
    <div data-magellan-expedition="fixed">
      <dl class="sub-nav">
        <dd data-magellan-arrival="place1"><a href="#place1">高雄市</a></dd>
        <dd data-magellan-arrival="place2"><a href="#place2">台北市</a></dd>
      </dl>
    </div>

    <p style="height: 200px"></p>

    <a name="place1"></a>
    <h3 data-magellan-destination="place1">高雄市</h3>
    <p>___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___
      ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___ ___
      ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___ ___
      ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ </p>
```

```
    <p style="height: 400px"></p>

    <a name="place2"></a>
    <h3 data-magellan-destination="place2">台北市</h3>
    <p>___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___
     ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___
     ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
___ ___
     ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ </p>

    <p style="height: 1000px"></p>
```

上述动态置顶追踪型导航栏主要由如下结构所共同构成。

- 第一层有如下多种元素。
  - 首先是 data-magellan-expedition 属性值为 fixed 的 div 元素，并且内含 class 属性值为 sub-nav 的 dl 元素。在此 dl 元素中，还含有 data-magellan-arrival 属性值为 place 1 或 place2 的 dd 元素。
  - name 属性值为 place1 的 a 元素。
  - data-magellan-destination 属性值为 place1 的 h1 至 h6 元素之一。
  - name 属性值为 place2 的 a 元素。
  - data-magellan-destination 属性值为 place2 的 h1 至 h6 元素之一。
  - 用来撑高网页高度的多个 p 元素，以便于读者进行范例程序的测试。

5.7 子导航栏

本节的程序代码都来自于 Ch05-07-sub-nav.html 范例文件。

在图 5-23 中，可以看到所谓的**子导航栏**里有一个标题文字（筛选：）和 4 个超链接（所有、最近一年、最近一个月、最近一周）。

图 5-23 子导航栏

若浏览器的窗口宽度不足时，【最近一周】超链接就会如图 5-24 所示，被隐藏起来而不显示。

图 5-24 浏览器的窗口宽度不足时【最近一周】超链接就会被隐藏起来

```
<dl class="sub-nav">
   <dt>筛选：</dt>

   <dd><a href="#">所有</a></dd>

   <dd class="active"><a href="#">最近一年</a></dd>

   <dd><a href="#">最近一个月</a></dd>

   <dd class="hide-for-small-only"><a href="#">最近一周</a></dd>
</dl>
```

上述子导航栏主要由如下结构所共同构成。

- 第一层是 class 属性值为 sub-nav 的 dl 元素。
- 第二层除了有 dt 元素之外，还有 dd 元素。其中，有些 dd 元素的 class 属性值可能会被应用如下不同的 CSS 规则名称。
  - ➢ 当 class 属性值为 active 时，使得此 dd 元素显示【处于活动中】状态的外观。
  - ➢ 当 class 属性值为 hide-for-small-only ，而且浏览器的窗口宽度变成小型屏幕尺寸的宽度时，则此 dd 元素在画面中会被隐藏起来。

5.8 浏览分层提示

本节的程序代码都来自于 Ch05-08-breadcrumbs.html 范例文件。

在图 5-25 中，存在两组看起来一样的浏览分层提示结构。其中，第一组浏览分层提示结构是由 ul 元素和内部的 li 元素所构成的，第二组则是由 nav 元素和内部的 li 元素所构成的。

台湾 / 台北市 / 万华区 / 龙山寺

台湾 / 台北市 / 万华区 / 龙山寺

图 5-25 浏览分层提示

```
<ul class="breadcrumbs">
   <li><a href="#">台湾</a></li>
   <li><a href="#">台北市</a></li>
   <li class="unavailable"><a href="#">万华区</a></li>
   <li class="current"><a href="#">龙山寺</a></li>
</ul>

<nav class="breadcrumbs">
   <li><a href="#">台湾</a></li>
   <li><a href="#">台北市</a></li>
   <li class="unavailable"><a href="#">万华区</a></li>
   <li class="current"><a href="#">龙山寺</a></li>
</nav>
```

上述浏览分层提示主要由如下结构所共同构成。

- 第一层是 class 属性值为 breadcrumbs 的 ul 或 nav 元素。
- 第二层是 li 元素，它的 class 属性值可能会被应用如下不同的 CSS 规则名称：
  - class 属性值为 unavailable 时，此 li 元素会显示出【停用中】状态的外观。
  - class 属性值为 current 时，此 li 元素会显示出【使用中】状态的外观。

5.9 分页导航栏

本节的程序代码都来自于 Ch05-09-pagination.html 范例文件。

在图 5-26 中的分页导航栏里，除了有最左侧的【前一页 <<】按钮和最右侧的【下一页 >>】按钮之外，还可设计出第 1 页至第 13 页的导航按钮。

图 5-26 分页导航栏

```
<div class="pagination-centered">
  <ul class="pagination">

    <li class="arrow unavailable"> <a href="">&laquo;</a></li>

    <li class="current"><a href="">1</a></li>

    <li><a href="">2</a></li>
    <li><a href="">3</a></li>
    <li><a href="">4</a></li>

    <li class="unavailable"><a href="">&hellip;</a></li>

    <li><a href="">12</a></li>
    <li><a href="">13</a></li>

    <li class="arrow"><a href="">&raquo;</a></li>

  </ul>
</div>
```

上述分页导航栏主要由如下结构所共同构成。

- 第一层是 class 属性值为 pagination-centered 的 div 元素。
- 第二层是 class 属性值为 pagination 的 ul 元素。
- 第三层是 li 元素，它的 class 属性值可能会应用如下不同的 CSS 规则名称。

- class 属性值为 arrow unavailable 时，此 li 元素会显示出带有【箭头符号】且【停用中】状态的外观。
- class 属性值为 current 时，此 li 元素会显示出【使用中】状态的外观。

5.10 缩略图

本节的程序代码都来自于 Ch05-10-thumbnails.html 范例文件。

当鼠标指针移到图 5-27 所示的**缩略图**超链接的范围内时，缩略图默认会出现淡蓝色的**边框**，此时鼠标指针也变成手掌的形状。

图 5-27 图片出现淡蓝色的边框同时鼠标变成手掌形状

在网页中，选择如上的缩略图超链接，可以看到如图 5-28 所示的大型照片。

图 5-28 选择缩略图后可以看到大型的照片

如上的大型照片**再被选中一次**，即可得到如图 5-29 所示的原始尺寸的照片。

图 5-29　原始尺寸的照片

此时为了回到原来的页面，就必须在浏览器中单击如图 5-30 所示的【上一页】按钮。

图 5-30　单击【上一页】按钮

回到原来的页面之后，可看到如图 5-31 所示的画面。

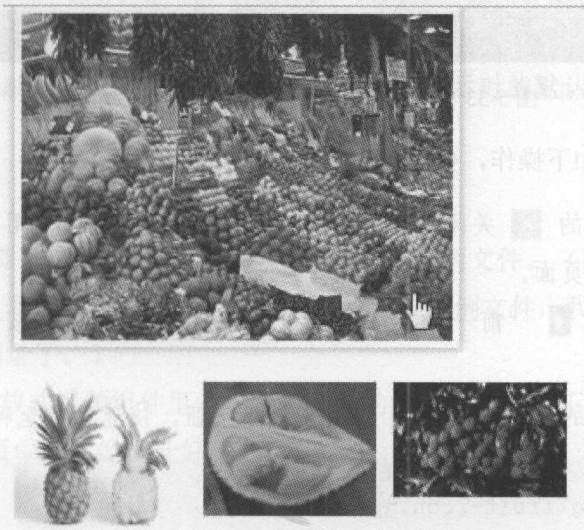


图 5-31　回到原来的画面

在图 5-32 所示的图中单击其中一张照片的缩略图，可以看到其对应的大照片。

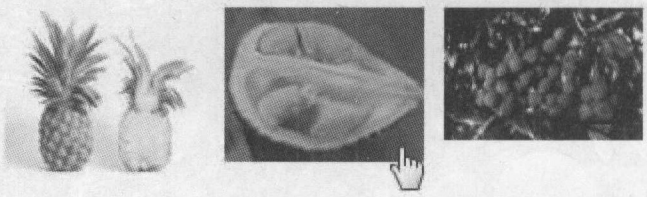


图 5-32　单击其中一张照片的缩略图

单击图 5-32 中的缩略图之后，可以看到其对应的如图 5-33 所示的大型照片，在页面当中，我们还可以发现提供给用户操作的一些额外工具按钮。

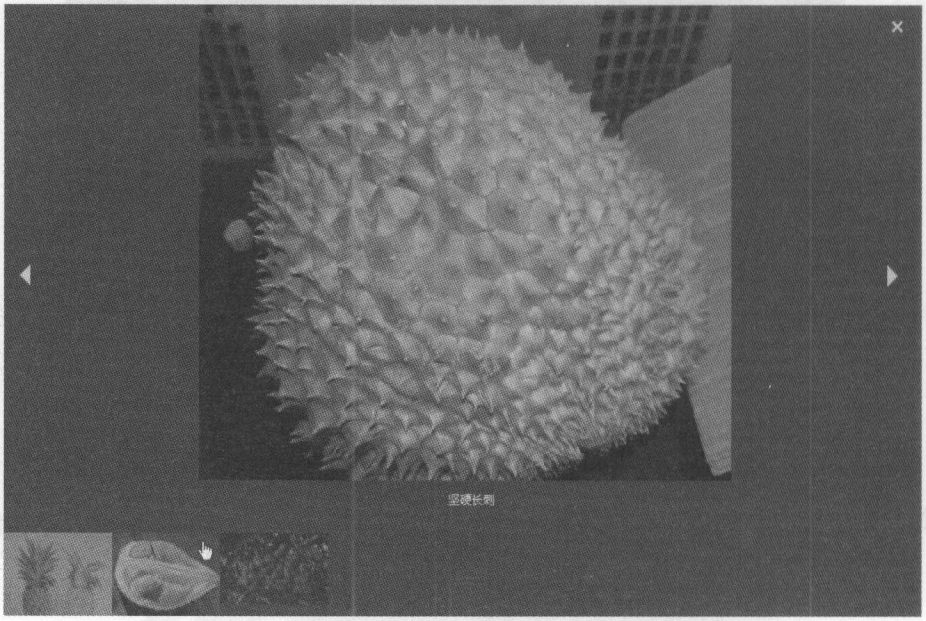


图 5-33　单击缩略图可以看到其对应的大型照片

在上述页面进行如下操作，会有对应的功能。

- 可单击右上角的 × 关闭按钮或键盘上的【Esc】键，以关闭当前的大型照片页面，而回到原来的页面。
- 可单击左侧的 ◀ “前一张”按钮或右侧的 ▶ “下一张”按钮，以切换不同的大型照片。
- 可单击左下角的  缩略图分组按钮，快速切换至特定的大型照片。

```
<a class="th" href="img/fruit.jpg">
   <img src="img/fruit-icon.jpg">
</a>
```

```
    <p></p>

      <ul class="clearing-thumbs" data-clearing>

        <li>
          <a href="img/pineapple.jpg"><img data-caption="剖面图"
src="img/pineapple
            -icon.jpg" width="100" height="100"></a>
        </li>

        <li>
          <a href="img/durian.jpg"><img data-caption="坚硬长刺" src="img/
            durian-icon.jpg" width="100" height="100"></a>
        </li>

        <li>
          <a href="img/litchi.jpg"><img data-caption="一大串果实" src="img/
            litchi-icon.jpg" width="100" height="100"></a>
        </li>

      </ul>
```

上述缩略图主要由如下结构所共同构成。

第一层有如下几种元素。

- class 属性值为 th、其 href 属性值为图片文件路径的 a 元素，并内含一个 img 元素，它的 src 属性值为同一张照片的较小版本的图片文件路径。
- class 属性值为 clearing-thumbs 并带有无属性值的 data-clearing 属性的 ul 元素，并内含如下子元素。
  href 属性值为图片文件路径的 a 元素，并内含一个 img 元素，其 data-caption 属性值为特定的说明文字，其 src 属性值为同一张照片的较小版本的图片文件路径，并通过 width 和 height 属性来设置此照片在显示时的默认宽度与高度。

5.11 缩放式视频

本节的程序代码都来自于 Ch05-11-flex-video.html 范例文件。（改编者注：如果你的系统不能访问YouTube，可以参照Ch05-11-flex-video01.html范例文件，我们把YouTube的网址换成Youku（优酷）的网址了。）

在图 5-34 中的缩放式视频组件里，左上角开始的标题文字和底下的播放视频的工具按钮其实在优酷的原始网页中就已经设计好了。

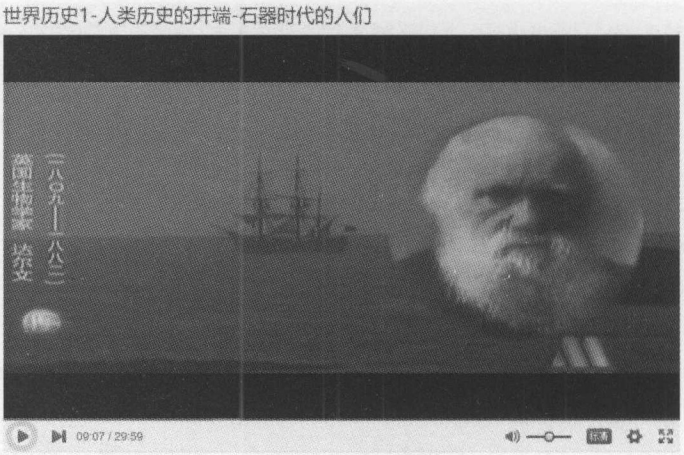


图 5-34　缩放式视频

```
<div class="flex-video widescreen vimeo">
   <iframe src=" http://v.youku.com/v_show/id_XNzA0ODYyMTY=.html?f=5517394">
   </iframe>
</div>
```

上面的组件主要由 class 属性值为 flex-video widescreen vimeo的 div 元素，并内含 src 属性值为特定视频网址的 iframe 元素所构成。

5.12　窗体组件

本节的程序代码都来自于 Ch05-12-forms.html 范例文件。

在图 5-35 中的窗体组件包括标签（手机号码或账号名称、住宅地址、联络电话等）、文本栏、单选按钮（较甜的、较酸的）、多选按钮（份量加大、增加配料）和多行文本栏。

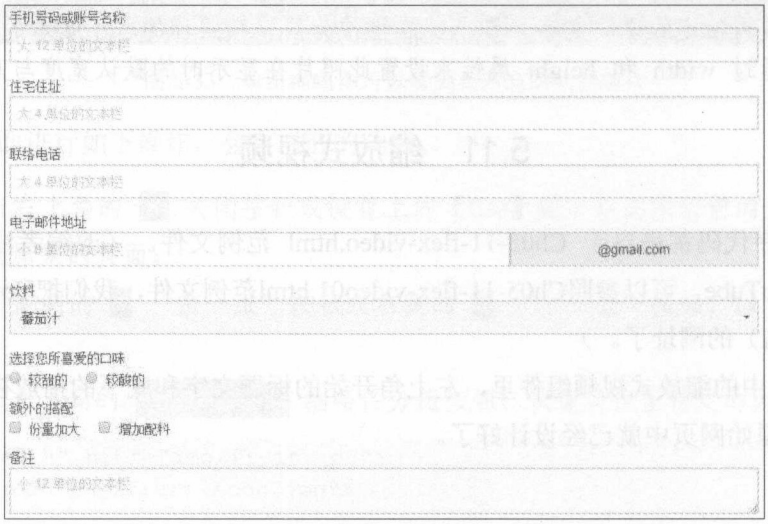


图 5-35　窗体组件

在图 5-36 中的部分窗体组件里，可以看到每个标签都显示在对应的文本栏的左上方，而在各个文本栏的范围内都有浅灰色的提示文字。

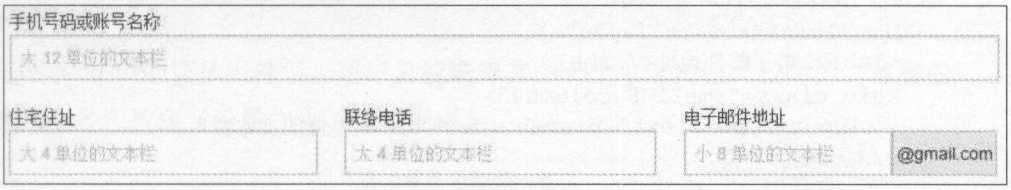


图 5-36　窗体中的文本栏

当浏览器的窗口宽度不足时，图 5-35 中的窗体组件会被纵向排列成为如图 5-37 所示的模样。

图 5-37　当浏览器的窗口宽度不足时窗体中的组件被纵向排列

```
<form>
  <div class="row">
    <div class="large-12 columns">
      <label>手机号码或账号名称
        <input type="text" placeholder="大 12 单位的文本栏" />
      </label>
    </div>
  </div>

  <div class="row">
    <div class="large-4 columns">
      <label>住宅地址
        <input type="text" placeholder="大 4 单位的文本栏" />
      </label>
    </div>

    <div class="large-4 columns">
      <label>联络电话
        <input type="text" placeholder="大 4 单位的文本栏" />
```

```
        </label>
      </div>

      <div class="large-4 columns">
        <div class="row collapse">
          <label>电子邮件地址</label>
          <div class="small-8 columns">
            <input type="text" placeholder="小 8 单位的文本栏" />
          </div>

          <div class="small-4 columns">
            <span class="postfix">@gmail.com</span>
          </div>

        </div>
      </div>
    </div> <!-- row end -->
```

在图 5-38 所示的传统**下拉式列表**组件里，默认会占用**上层元素**的**总宽度**。当用户在其范围内**单击一下**时，便可打开下拉式列表。

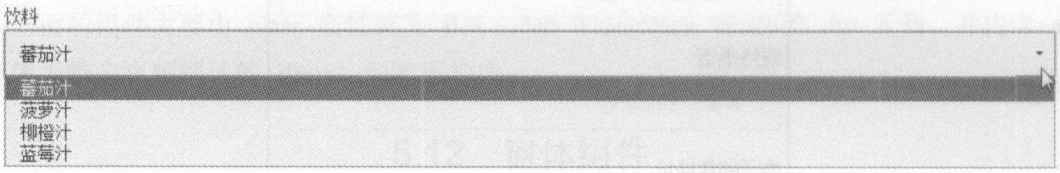


图 5-38　打开下拉式列表

浏览器的窗口宽度越来越小时，此下拉式列表也会越变越窄，如图 5-39 所示。

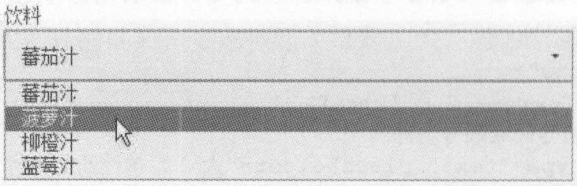


图 5-39　浏览器的窗口宽度变窄时下拉式列表随之变窄

```
<div class="row">

  <div class="large-12 columns">

    <label>饮料
      <select>
        <option value="tomato">番茄汁</option>
        <option value="pineapple">菠萝汁</option>
        <option value="orange">柳橙汁</option>
        <option value="blueberry">蓝莓汁</option>
      </select>
    </label>
```

```
    </div>

  </div>
```

在图 5-40 中的窗体组件里，用户**只能选择其中一个**单选按钮（较甜的、较酸的），但可以**勾选几个**多选按钮（份量加大、增加配料）。

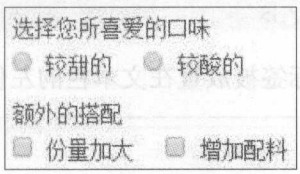


图 5-40 窗体组件中的单选按钮和多选按钮

```
<div class="row">

  <div class="large-6 columns">
    <label>选择您所喜爱的口味</label>

    <input type="radio" name="pokemon" value="sweet" id="pokemonRed">
    <label for="sweet">较甜的</label>
    <input type="radio" name="pokemon" value="salty" id="pokemonBlue">
    <label for="salty">较酸的</label>
  </div>

  <div class="large-6 columns">
    <label>额外的搭配</label>

    <input id="double" type="checkbox">
    <label for="double">份量加大</label>
    <input id="spicy" type="checkbox">
    <label for="spicy">增加配料</label>
  </div>

</div>
```

在图 5-41 中的备注栏是**多行**文本栏，可供用户输入多行的内容，并允许用户**拖曳**此字段栏右下角的 **缩放按钮**。

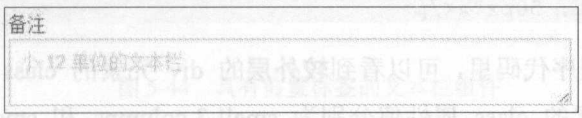


图 5-41 备注栏是多行文本栏

```
  <div class="row">

    <div class="large-12 columns">
      <label>备注
```

```
        <textarea placeholder="小 12 单位的文本栏"></textarea>
      </label>
    </div>

  </div>

</form>

<p style="height: 50px"></p>
```

在图 5-42 中的窗体组件里，标签被放置在文本栏的左侧。

图 5-42　标签被放置在文本栏的左侧

```
<form>

  <div class="row">

    <div class="small-8">

      <div class="row">

        <div class="small-3 columns">
          <label for="right-label" class="right inline">行内标签文字</label>
        </div>

        <div class="small-9 columns">
          <input type="text" id="right-label" placeholder="行内文本栏">
        </div>

      </div>

    </div>

  </div>

</form>

<p style="height: 50px"></p>
```

在上述范例中的程序代码里，可以看到**较外层**的 div 元素的 class 属性值为 small-8，而**最内层**的两个 div 元素的 class 属性值分别为 small-3 columns 和 small-9 columns，也就是：

- 在小尺寸屏幕中，较外层的 div 元素会占用上层元素的 12 等分当中的 8 等分宽度，也就是其三分之二（8 / 12）的上层元素的宽度。

- 在小尺寸屏幕中，最内层的两个 div 元素会分别占用上层元素的 12 等分当中的 3 等分和 9 等分，也就是分别为四分之一（3 / 12）和四分之三（9 / 12）的上层元素的宽度。

在图 5-43 中的窗体组件里，利用 fieldset 元素来包装可构成一组的部分窗体组件。在此，fieldset 元素只内含了一个 label 元素和 type 属性值为 text 的 input 元素而已。

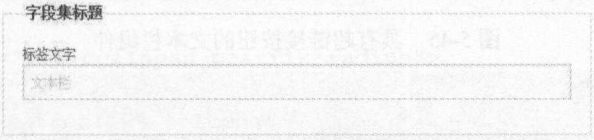


图 5-43 构成一组的窗体组件

```
<form>

  <fieldset>
```

- **<legend>字段集标题</legend>**

```
    <label>标签文字
      <input type="text" placeholder="文本栏">
    </label>

  </fieldset>

</form>

<p style="height: 50px"></p>

<form>

  <div class="row collapse">
```

在图 5-44 的窗体组件里，通过看起来比 label 元素打造的标签更具质感的 class 属性值为 prefix 的 span 元素标签来搭建整体成形的文本栏组件。

图 5-44 具有前置标签的文本栏组件

```
<div class="small-3 large-2 columns">
  <span class="prefix">http://</span>
</div>

<div class="small-9 large-10 columns">
  <input type="text" placeholder="请输入网址 ...">
```

```
    </div>

  </div>
```

在图 5-45 中的窗体组件里，左侧为一个文本栏，右侧为一个 class 属性值为 **button postfix** 的超链接 a 元素所打造出来的按钮。

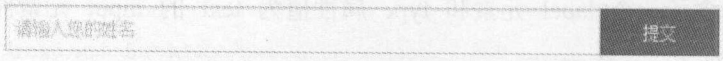


图 5-45　具有超链接按钮的文本栏组件

```
<div class="row">

  <div class="large-12 columns">

    <div class="row collapse">

      <div class="small-10 columns">
        <input type="text" placeholder="请输入您的姓名">
      </div>

      <div class="small-2 columns">
        <a href="#" class="button postfix">提交</a>
      </div>

    </div>

  </div>

</div>
```

在图 5-46 中，窗体组件是由 class 属性值为 **prefix** 或 **postfix** 的 span 元素搭配文本栏所打造出来的。

图 5-46　具有前置和后置链接按钮的文本栏组件

```
<div class="row">

  <div class="large-6 columns">

    <div class="row collapse prefix-radius">

      <div class="small-3 columns">
        <span class="prefix">居住城市</span>
      </div>
```

```
        <div class="small-9 columns">
          <input type="text" placeholder="请输入城市名称">
        </div>

      </div>

    </div>

    <div class="large-6 columns">

      <div class="row collapse postfix-radius">

        <div class="small-9 columns">
          <input type="text" placeholder="出生日期的格式：yyyy-mm-dd">
        </div>

        <div class="small-3 columns">
          <span class="postfix">出生日期</span>
        </div>

      </div>

    </div>

  </div>
```

在本范例的程序代码结构里，**prefix** 有**前置**的意思，而 **postfix** 则有**后置**的意思。而 **prefix-radius** 或 **postfix-radius** 则都会使整体元素带有**圆角矩形**的**顶点**样式。

在图 5-47 中，窗体组件是由 class 属性值为 **button prefix** 或 **button postfix** 的 a 元素所搭建的按钮和文本栏所组成的。

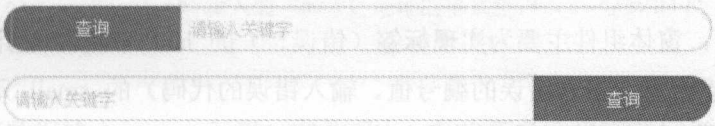


图 5-47 具有前置和后置按钮的文本栏组件

```
<div class="row">

  <div class="large-6 columns">

    <div class="row collapse prefix-round">

      <div class="small-3 columns">
        <a href="#" class="button prefix">查询</a>
      </div>

      <div class="small-9 columns">
        <input type="text" placeholder="请输入关键字">
```

```
        </div>

      </div>

    </div>

    <div class="large-6 columns">

      <div class="row collapse postfix-round">

        <div class="small-9 columns">
          <input type="text" placeholder="请输入关键字">
        </div>

        <div class="small-3 columns">
          <a href="#" class="button postfix">查询</a>
        </div>

      </div>

    </div>

  </div>

  </form>

  <p style="height: 50px"></p>

  <form>
```

在本范例的程序代码结构里，**prefix-round** 或 **postfix-round** 都会使整体元素带有**圆角矩形**的**顶点**样式。

在图 5-48 中，窗体组件主要为**实现标签**（错误！）的 label 元素、**让用户输入内容**的文本栏以及**显示错误信息**（输入错误的编号值、输入错误的代码）的 small 元素。标签文字带有特殊颜色，而错误信息则是**白色斜体字**加上**特殊背景**。

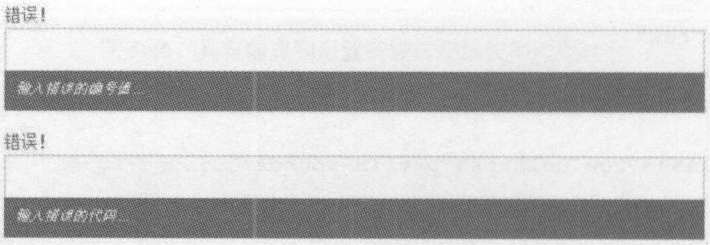


图 5-48　实现报错标签的文本栏组件和显示错误信息的文本栏组件

```
  <div class="row">

    <div class="large-6 columns">
```

```
    <label class="error">错误!
      <input type="text" class="error" />
    </label>

    <small class="error">输入错误的编号值 ...</small>
  </div>

  <div class="large-6 columns error">
    <label>错误!
      <input type="text" />
    </label>

    <small class="error">输入错误的代码 ...</small>

  </div>

</div>
```

在上述范例的程序代码中，主要是通过 class="error" 的语句使得标签文字或错误信息有了特殊的字体样式。

在图 5-49 中，没有标签，但文本栏是由可输入多行内容的 textarea 元素所搭建而成的。

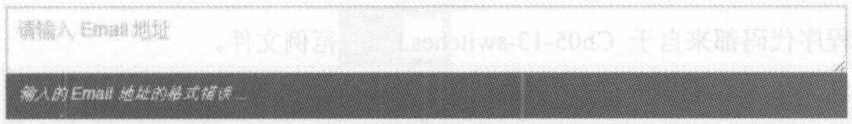


图 5-49 可以输入多行内容的文本栏组件

```
  <textarea class="error" placeholder="请输入 Email 地址"></textarea>

  <small class="error">输入的 Email 地址的格式错误 ...</small>

</form>
```

可输入多行数据的 textarea 元素，在外观上，其右下角会出现鼠标指针可拖曳而缩放其尺寸的 缩放控制按钮。

本节的**窗体**和其内部的各个组件主要由如下结构所共同组成。

- 第一层是 form 元素。
- 第二层有几个 class 属性值为 row 的 div 元素，并内含如下子元素。
  - 内部第一层是一个或几个 div 元素，其 class 属性值为 large-12 columns、large-column-4、large-column-6、small-8、small-3 columns、small-9 columns、small-3 large-2 columns、small-9 large-10 columns、small-10 columns、small-2 columns 或 large-6 columns error。其中，large-12 columns 是指此元素的宽度为窗口宽度的 12

等分中的 12 等分，也就是**整个窗口宽度（12 / 12）**，而 large-4 columns 是指此元素的宽度为窗口宽度的 12 等分中的 4 等分，也就是窗口宽度的**三分之一（4 / 12）**。

- 内部各个层次，则包含如下一部分的元素。
  - ◊ label 元素。
  - ◊ type 属性值为 text 的 input 元素（文本栏）。
  - ◊ class 属性值为 postfix 的 span 元素。
  - ◊ select 元素与内层的 option 元素。
  - ◊ type 属性值为 checkbox 的 input 元素（多选按钮 / 复选框）。
  - ◊ textarea 元素。
  - ◊ fieldset 元素和内层的 legend 元素。
  - ◊ class 属性值为 row collapse、row collapse prefix-radius、row collapse postfix-radius、row collapse prefix-round 或 row collapse postfix-round 的 div 元素。
  - ◊ class 属性值为 button prefix 或 button postfix 的 button 元素。

5.13 切换按钮

本节的程序代码都来自于 Ch05-13-switches.html 范例文件。

在图 5-50 中，可以看到有 4 个切换按钮。第 1 个切换按钮是自成一组的，而第 2 个至第 4 个切换按钮，虽然在外观上不相同，但是却被设置成为一组。换句话说，在同一时间内，**只能有其中一个切换按钮**可以启用。

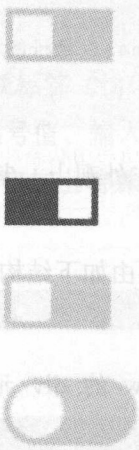


图 5-50 切换按钮

将鼠标指针移到第 1 个切换按钮的范围之内，单击一下之后即可启用此切换按钮，其外观也会从浅灰色（见图 5-51）变成默认的蓝色。

图 5-51 启用前切换按钮显示为灰色

如上的切换按钮被启用之后，外观变成如图 5-52 所示的状态。

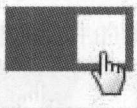


图 5-52 启用后切换按钮显示为蓝色

```
<div class="switch">
   <input id="exampleCheckboxSwitch" type="checkbox">
   <label for="exampleCheckboxSwitch"></label>
</div>

<p style="height: 20px"></p>
```

图 5-53 中的 3 个造型不太一样的切换按钮其实是一组的。刚被选中的那个切换按钮会变成**启用**的状态，而其他两个则会变成**被关闭**的状态。

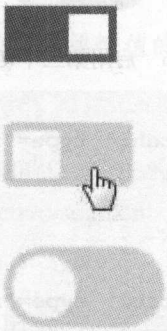


图 5-53 一组切换按钮启用状态和关闭状态时的显示

如果选中第二个切换按钮，就可以看到第二个切换按钮**被启用**了，如图 5-54 所示。

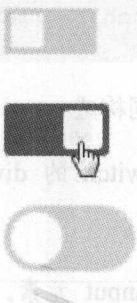


图 5-54 第二个切换按钮被启用了

在图 5-55 中单击第三个切换按钮，可看到第三个切换按钮**被启用**了，如图 5-56 所示。

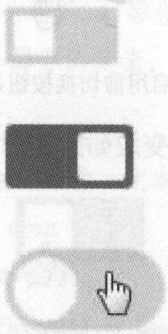


图 5-55　准备启用第三个切换按钮

图 5-56　启用第三个切换按钮

```
<div class="switch small">
   <input id="exampleRadioSwitch1" type="radio" checked name="testGroup">
   <label for="exampleRadioSwitch1"></label>
</div>

<div class="switch radius">
   <input id="exampleRadioSwitch2" type="radio" name="testGroup">
   <label for="exampleRadioSwitch2"></label>
</div>

<div class="switch round large">
   <input id="exampleRadioSwitch3" type="radio" name="testGroup">
   <label for="exampleRadioSwitch3"></label>
</div>
```

上述切换按钮主要由如下结构所共同构成。

- 第一层有几个 class 属性值为 switch 的 div 元素。
- 第二层有如下元素。
  - type 属性值为 checkbox 的 input 元素。
  - for 属性值为上述 input 元素的 id 属性值的 label 元素。

在本范例中，第 3 个切换按钮的边框有**圆角矩形**样式，而第 4 个切换按钮的边框有**椭圆形**样式，这是由于它们分别受到 class 属性值内有 radius 和 round 的影响。

5.14 滑杆

本节的程序代码都来自于 Ch05-14-range-slider.html 范例文件。

在图 5-57 中，可通过鼠标指针拖曳滑杆组件的**滑动方块（简称滑块）**来增加（向右）或减少（向左）其对应的数值。

图 5-57 横向滑杆

```
<div class="range-slider" data-slider>
   <span class="range-slider-handle" role="slider" tabindex="0"></span>
   <span class="range-slider-active-segment"></span>
   <input type="hidden">
</div>
```

上述滑杆主要由如下结构所共同构成。

class 属性值为 range-slider，而且带有无属性值的 data-slider 属性的 div 元素，其内层包含如下子元素。

- class 属性值为 range-slider-handle、role 属性值为 slider 的 span 元素。
- class 属性值为 range-slider-active-segment 的 span 元素。
- type 属性值为 hidden 的 input 元素。

在图 5-58 所示的滑杆组件中，可**纵向**拖曳其滑块来增加（向上）或减少（向下）其对应的数值。

图 5-58 纵向滑杆

```
    <div class="range-slider vertical-range" data-slider data-options="vertical:
true;">
      <span class="range-slider-handle" role="slider" tabindex="0"></span>
      <span class="range-slider-active-segment"></span>
      <input type="hidden">
    </div>
```

上述滑杆主要由如下结构所共同构成。

class 属性值为 range-slider vertical-range、data-options 属性值为 vertical: true 并带有无属性值的 data-slider 属性的 div 元素，其内层包含如下子元素。

- class 属性值为 range-slider-handle、role 属性值为 slider 的 span 元素。
- class 属性值为 range-slider-active-segment 的 span 元素。
- type 属性值为 hidden 的 input 元素。

在图 5-59 所示的滑杆组件中，可以看到右侧显示滑杆所对应的数值（77），以明确让用户知道目前设置的数值是多少。

图 5-59　同时显示滑杆设置的数值

```
<div class="row">

    <div class="small-10 medium-11 columns">
      <div class="range-slider" data-slider data-options="display_selector:
        #sliderOutput3;">
        <span class="range-slider-handle" role="slider" tabindex="0"></span>
        <span class="range-slider-active-segment"></span>
      </div>
    </div>

    <div class="small-2 medium-1 columns">
      <span id="sliderOutput3"></span>
    </div>

</div>
```

上述带设置数值的滑杆主要由如下结构所共同构成。

class 属性值为 row 的 div 元素，并内含如下子元素。

- class 属性值为 small-10 medium-11 columns 的 div 元素，并内含class 属性值为 range-slider、data-options 属性值为 display_selector: #sliderOutput3 而且带有无属性值的 data-slider 属性的 div 元素（内含两个span元素）。

◊ class 属性值为 range-slider-handle、role 属性值为 slider 的 span 元素。
◊ class 属性值为 range-slider-active-segement 的 span 元素。

➢ class 属性值为 small-2 medium-1 columns 的 div 元素，并内含 id 属性值为 sliderOutput3 的 span 元素。

在本范例的程序代码结构里，为了让滑杆和显示数值的元素互动，在滑杆的 div 元素语句上，要写上 **data-options="display_selector: #sliderOutput3;"**，而在显示数值的 span 元素上，要写上 **id="sliderOutput3"**。如此一来，滑杆的当前操作数值才能反应到 id 为 sliderOutput3 的 span 元素上。

在图 5-60 中的滑杆组件里，右侧有个数值型（number）的文本栏，会反映出滑杆当前被拖曳的程度。但是未加以改版之前，用户直接在右侧数值型文本栏内，输入新的数值再按下 Enter 键，并没办法让滑块的位置移动到右侧数值所指定的位置。

图 5-60 通过数值型文本栏输入数字来调整滑杆滑块的设置

```
<div class="row">

  <div class="small-10 columns">

    <div class="range-slider" data-slider data-options="display_selector:
      #days-off-count; initial: 28;">

      <span class="range-slider-handle" role="slider" tabindex="0"></span>

      <span class="range-slider-active-segment"></span>

    </div>

  </div>

  <div class="small-2 columns">
    <input type="number" id="days-off-count" value="28" />
  </div>

</div>

<p style="height: 50px"></p>
```

上述带数值型文本栏的滑杆主要由如下结构所共同构成。

class 属性值为 row 的 div 元素，并内含如下子元素。

➢ class 属性值为 small-10 columns 的 div 元素，并内含class 属性值为 range-slider、data-options 属性值为 display_selector: #days-off-count; initial: 28 而且带有无属性

值的 data-slider 属性的 div 元素（内含两个span元素）。

◊ class 属性值为 range-slider-handle、role 属性值为 slider 的 span 元素。
◊ class 属性值为 range-slider-active-segement 的 span 元素。

➢ class 属性值为small-2 columns 的 div 元素，并内含 type 属性值为 number、id 属性值为 days-off-count、value 属性值为 28 的 input 元素。

在本范例的程序代码结构中，可让滑杆一开始就显示在网页当中，通过 data-options="display_selector: #days-off-count; **initial: 28;**"，确定其滑块位于刻度 28 的位置上，并且将其滑块所在位置的刻度数值反映到 id 属性值为 days-off-count、type 属性值为 number 的 input 元素上。

在图 5-61 中的滑杆组件里，组件本身或内部的滑块边框都是**圆角矩形**的样式。

图 5-61　具有圆角的滑杆和滑块

```
<div class="range-slider radius" data-slider>
   <span class="range-slider-handle" role="slider" tabindex="0"></span>
   <span class="range-slider-active-segment"></span>
   <input type="hidden">
</div>
```

在本程序代码结构中，div 元素的 class 属性值内含 radius 字样，使得其具有**圆角矩形**的样式。

在图 5-62 中的滑杆组件里，组件本身或内部的滑块边框都是**椭圆形**的样式。

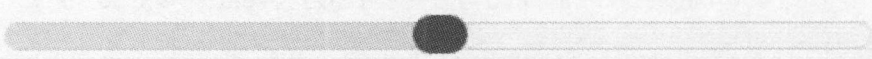


图 5-62　具有椭圆角的滑杆和滑块

```
<div class="range-slider round" data-slider>
   <span class="range-slider-handle" role="slider" tabindex="0"></span>
   <span class="range-slider-active-segment"></span>
   <input type="hidden">
</div>
```

上述具有椭圆角的滑杆和滑块主要由如下结构所共同构成。

class 属性值为 range-slider round 的 div 元素，并内含如下子元素。

➢ class 属性值为 range-slider-handle、role 属性值为 slider 的 span 元素。
➢ class 属性值为 range-slider-active-segment 的 span 元素。
➢ type 属性值为 hidden 的 input 元素。

在本程序代码结构中，div 元素的 class 属性值内含 round 字样，使得其具有**椭圆形**的样式。

在图 5-63 中的滑杆组件里，可发现有**禁止**符号在滑块附近，即用户暂时无法拖曳此滑块。

图 5-63 滑块被禁止拖曳时的情况

```
<div class="range-slider disabled" data-slider disabled>
  <span class="range-slider-handle" role="slider" tabindex="0"></span>
  <span class="range-slider-active-segment"></span>
  <input type="hidden">
</div>

<p style="height: 50px"></p>
```

滑块被禁止拖曳时的滑杆组件主要由如下结构所共同构成。

class 属性值为 range-slider disabled，而且带有两个无属性值的 data-slider 和 disabled 属性的 div 元素，并内含如下子元素。

- class 属性值为 range-slider-handle、role 属性值为 slider 的 span 元素。
- class 属性值为 range-slider-active-segment 的 span 元素。
- type 属性值为 hidden 的 input 元素。

在本范例的程序代码结构中，受到 class 属性值带有 disabled 字样的影响，让用户暂时无法操作此滑杆。要恢复成为class 属性值不带有 disabled 字样，则需要程序设计人员额外通过 JavaScript 程序代码来实现。

在图 5-64 中的滑杆组件里，拖曳其滑块**并不顺畅**！其实是因为此滑杆设置了一次要**跳动 20 个**单位刻度，所以水平拖曳滑块一段距离之后，才会看到滑块的跳动。

图 5-64 设置了让滑块一次要跳动 20 个单位刻度

```
<div class="range-slider" data-slider data-options="step: 20;">
  <span class="range-slider-handle" role="slider" tabindex="0"></span>
  <span class="range-slider-active-segment"></span>
  <input type="hidden">
</div>

<p style="height: 50px"></p>
```

这个滑杆组件主要由如下结构所共同构成。

class 属性值为 range-slider，而且带有无属性值的 data-slider 属性的 div 元素，并内含如下子元素。

- class 属性值为 range-slider-handle、role 属性值为 slider 的 span 元素。
- class 属性值为 range-slider-active-segment 的 span 元素。
- type 属性值为 hidden 的 input 元素。

在本范例的程序代码结构中，为了让滑块每次被拖曳时都跳动 20 个单位刻度，必须设置其 div 元素的 data-options 属性值带有 **step: 20 ;** 的语句才行。

在如图 5-65 中的滑杆组件里，拖曳其滑块时，也会有不顺畅的情况，这是因为设置了其**滑动刻度**为本身宽度的 10 等分。

图 5-65　设置让滑块一次要跳动其本身宽度 10 等分的距离

```
<div class="range-slider" data-slider data-options="start: 1; end: 10;">
  <span class="range-slider-handle" role="slider" tabindex="0"></span>
  <span class="range-slider-active-segment"></span>
  <input type="hidden">
</div>
```

此滑杆组件主要由如下结构所共同构成。

class 属性值为 range-slider、data-options 属性值为 star1: 1 ; end: 10 ;，而且带有无属性值的 data-slider 属性的 div 元素，并内含如下子元素。

- class 属性值为 range-slider-handle、role 属性值为 slider 的 span 元素。
- class 属性值为 range-slider-active-segment 的 span 元素。
- type 属性值为 hidden 的 input 元素。

在本范例中，为了让滑杆组件刚好具有 10 个滑动刻度，必须在 div 元素上设置 data-options 属性值为 **start: 1 ; end: 10 ;**。

5.15　窗体验证

本节的程序代码都来自于 Ch05-15-validations.html 范例文件。

在图 5-66 中的部分窗体组件里，每个文本栏都要进行特定形式的**数据验证**。

在图 5-66 中的各个文本栏里，如果不输入资料就单击了【提交】按钮，那么各个文本栏所对应的**标签**除了变色之外，其**下方**还会出现**数据验证**相关的**错误信息**，如图 5-67 所示。

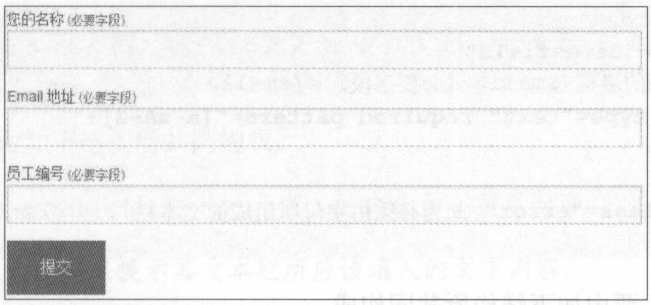


图 5-66　窗体中各个文本栏的验证

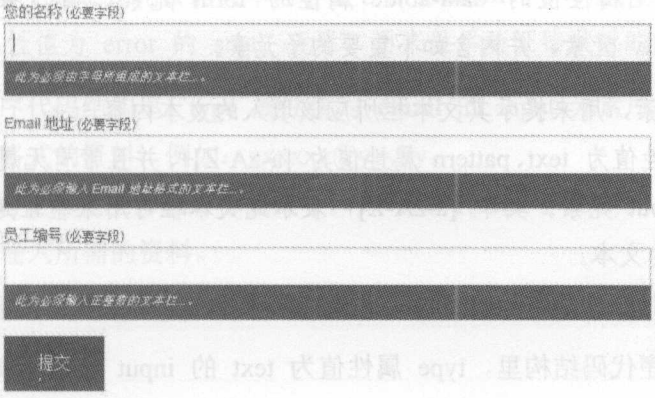


图 5-67　未通过数据验证则显示出相关的错误信息

在图 5-68 中的文本栏上方有标签文字，并写着【(必要字段)】，主要是用来提醒用户**必须**在此文本栏中输入所需的资料。

图 5-68　标签文字提示必须输入所需的资料

若图 5-68 中的文本栏内未输入任何资料，则发生**焦点**从该文本栏转移出去的如下情况。

- 闪烁的插入点被移到其他文本栏上。
- 【提交】按钮被按下。
- 用户在网页中单击了**不是超链接**的其他地方。

此时，该文本栏所对应的数据验证机制就会被启动而出现如图 5-69 所示的变化，使得其**标签**变色，并且出现数据验证相关的错误信息。

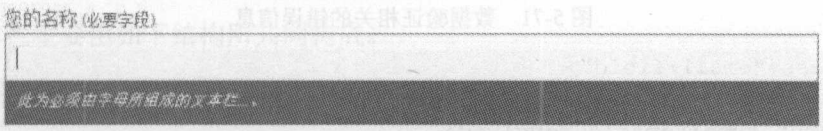


图 5-69　出现数据验证相关的错误信息

```
<form data-abide>
   <div class="name-field">
     <label>您的名称<small> (必要字段) </small>
       <input type="text" required pattern="[a-zA-Z]+">
     </label>

     <small class="error">此为必须由字母所组成的文本栏...。</small>
   </div>
```

此文本栏组件主要由如下结构所共同构成。

第一层为带有无属性值的 data-abide 属性的 form 元素，第二层为 class 属性值为 name-field 的 div 元素，并内含如下重要的子元素。

- label 元素，用来提示其文本栏所应该填入的文本内容。
- type 属性值为 text、pattern 属性值为 [a-zA-Z]+，并且带有无属性值的 required 属性的 input 元素。其中 [a-zA-Z]+ 表示此文本栏可用来验证**是否为大小写字母组合**而成的文本。
- class 属性值为 error 的 small 元素，用来显示验证错误时的信息。

在本范例的程序代码结构里，type 属性值为 text 的 input 元素所表示的文本栏写上了 pattern 属性值为 [a-zA-Z]+，表示此文本栏应该输入**至少一个大写**和**小写字母**（**[a-zA-Z]**）。其中，[a-zA-Z]+ 是一种**正则表达式**（regular expression）。

在图 5-70 中的文本栏上方也有标签文字，并写着【(必要字段)】，主要用来提醒用户**必须**在此文本栏中输入所需的资料。

图 5-70　标签文字提示必须输入所需的资料

若图 5-70 中的文本栏内未输入任何资料，就发生**焦点**从该文本栏转移出去的情况，那么该文本栏所对应的数据验证机制就会被启动，而出现如图 5-71 所示的变化，使得其标签变色，并且出现数据验证相关的错误信息。

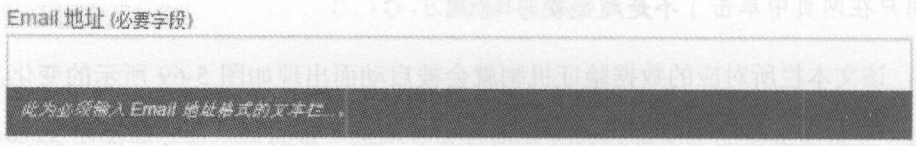


图 5-71　数据验证相关的错误信息

```
<div class="email-field">
   <label>Email 地址<small> (必要的字段)</small>
     <input type="email" required>
   </label>
```

```
    <small class="error">此为必须输入 Email 地址格式的文本栏...。</small>
</div>
```

此文本栏主要由如下结构所共同构成。

class 属性值为 email-field 的 div 元素，并内含如下重要元素。

- label 元素，用来提示其文本栏所应该填入的文字内容。
- input 元素，其 type 属性值为 email，而且带有无属性值的 required 属性。浏览器默认会去验证此元素的文本内容是否符合Email的书写格式。
- class 属性值为 error 的 small 元素，用来显示验证错误时的信息。

在本范例的程序代码结构里，type 属性值为 **email** 的 input 元素所表示的文本栏里应该输入电子邮件地址格式的资料，例如：xxxxxx@yyyyy。

在图 5-72 中的文本栏上方也有标签文字，并写着【(必要字段)】，主要是用来提醒用户**必须**在此文本栏中输入所需的资料。

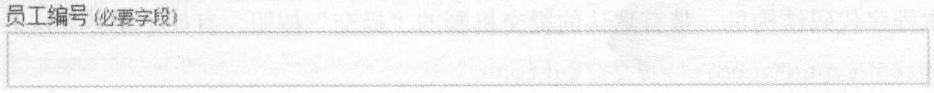


图 5-72 标签文字提示必须输入所需的资料

若图 5-72 中的文本栏内未输入任何资料，就会发生**焦点**从该文本栏转移出去的情况，那么该文本栏所对应的数据验证机制就会被启动，而出现如图 5-73 所示的变化，使得其标签变色，并且出现数据验证相关的错误信息。

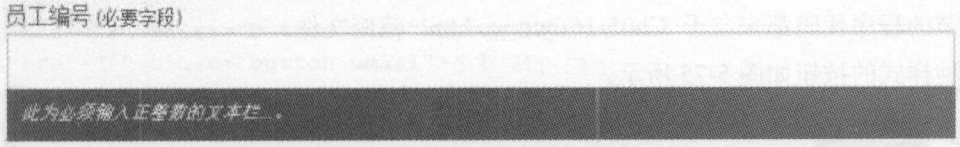


图 5-73 数据验证相关的错误信息

```
<div class="employee-no">
    <label>员工编号<small> (必要的字段)</small>
      <input type="text" required pattern="integer">
    </label>

    <small class="error">此为必须输入正整数的文本栏...。</small>
</div>
```

此文本栏主要由如下结构所共同构成。

class 属性值为 employee-no 的 div 元素，并内含如下重要的子元素。

- label 元素，用来提示其文本栏所应该填入的文字内容。

- input 元素，其 type 属性值为 text、pattern 属性值为 integer，而且带有无属性值的 required 属性。其中， integer 表示此文本栏会被浏览器验证是否为整数数值的数据内容。
- class 属性值为 error 的 small 元素，用来显示验证错误时的信息。

在本范例的程序代码结构里，type 属性值为 text 的 input 元素所表示的文本栏写上了 pattern 属性值为 **integer**，表示此文本栏应该输入**整数**（integer）数值类型的员工编号。

在图 5-74 中的按钮是由 type 属性值为 submit 的 button 元素所构成的。应用了 Zurb's Foundation 默认的按钮样式，才会有如此的外观。

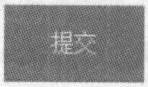


图 5-74 “提交”按钮

```
    <button type="submit">提交</button>
</form>
```

在本程序代码结构里，带有默认的**提交机制**的“提交”按钮，有如下两种写法。

```
<button type="submit">提交</button>
```

或

```
<input type="submit" value="提交">
```

5.16 按钮

本节的程序代码都来自于 Ch05-16-buttons.html 范例文件。

各种样式的按钮如图 5-75 所示。

图 5-75 各种样式的按钮

按钮元素主要是 class 属性值为 button 的 a 元素。其 class 属性值若额外加注如下的字

样，则会有不同的含义（图 5-76 为默认的按钮）。

- tiny，表示微小型的按钮。
- small，表示小型按钮。
- disabled，表示被停用的按钮。
- large，表示大型按钮。
- expand，表示扩展型（至窗口宽度）按钮。
- round，表示椭圆形按钮。
- radius，表示圆角矩形按钮。
- success、secondary、alert、info，表示应用不同颜色样式的按钮。

图 5-76　默认的按钮

```
<a href="#" class="button">默认的按钮</a>

<p style="height: 50px"></p>
```

图 5-77 为一般的普通按钮。

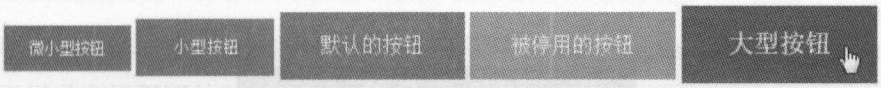


图 5-77　普通按钮

```
<!--尺寸类 -->
<a href="#" class="button tiny">微小型按钮</a>
<a href="#" class="button small">小型按钮</a>
<a href="#" class="button">默认的按钮</a>
<a href="#" class="button disabled">被停用的按钮</a>
<a href="#" class="button large">大型按钮</a>
```

图 5-78 为扩展式按钮。

图 5-78　扩展式按钮

```
<a href="#" class="button expand">扩展式按钮</a>
```

图 5-79 为圆角类按钮。

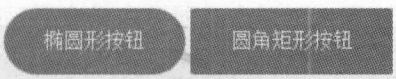


图 5-79　椭圆形按钮和圆角矩形按钮

```
<!--圆角类 -->
<a href="#" class="button round">椭圆形按钮</a>
<a href="#" class="button radius">圆角矩形按钮</a>

<p style="height: 50px"></p>
```

图 5-80 为各种颜色按钮。

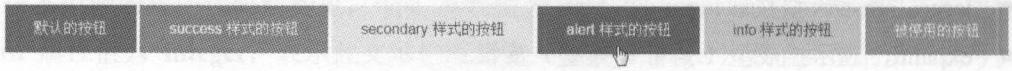


图 5-80 颜色类按钮

```
<!--颜色类 -->
<a href="#" class="button">默认的按钮</a>
<a href="#" class="button success">success 样式的按钮</a>
<a href="#" class="button secondary">secondary 样式的按钮</a>
<a href="#" class="button alert">alert 样式的按钮</a>
<a href="#" class="button info">info 样式的按钮</a>
<a href="#" class="button disabled">被停用的按钮</a>
```

5.17 按钮组

本节的程序代码都来自于 Ch05-17-button-groups.html 范例文件。

在图 5-81 中，可以看到一个按钮组内有 3 个应用了 Zurb's Foundation 样式的默认按钮，横向紧接在一起。

图 5-81 按钮组

```
<!--默认的按钮组 -->
<ul class="button-group">
   <li><a href="#" class="button">按钮 1</a></li>
   <li><a href="#" class="button">按钮 2</a></li>
   <li><a href="#" class="button">按钮 3</a></li>
</ul>

<p style="height: 30px"></p>
```

在本程序代码结构里，class 属性值为 button-group ，表示一个按钮组，而 class 属性值为 button ，表示一个按钮。

在图 5-82 中的按钮组里，内含 4 个紧接在一起的按钮。特别要注意**按钮组**本身的 4 个顶点的位置，它们都变成了**圆角**。

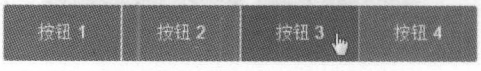


图 5-82 圆角矩形风格的按钮组

```
<!--圆角样式的按钮组 -->
<ul class="button-group radius">
   <li><a href="#" class="button">按钮 1</a></li>
   <li><a href="#" class="button">按钮 2</a></li>
   <li><a href="#" class="button">按钮 3</a></li>
   <li><a href="#" class="button">按钮 4</a></li>
</ul>
```

在本程序代码结构里，表示按钮组的 ul 元素语句中，其 class 属性值内含 radius 字样，使得此按钮组具有**圆角矩形**的风格。

图 5-83 中的按钮组变成了具有**椭圆形**样式的 4 个紧接在一起的按钮。

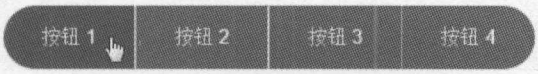


图 5-83 椭圆样式的按钮组

```
<!--椭圆样式的按钮组 -->
<ul class="button-group round">
   <li><a href="#" class="button">按钮 1</a></li>
   <li><a href="#" class="button">按钮 2</a></li>
   <li><a href="#" class="button">按钮 3</a></li>
   <li><a href="#" class="button">按钮 4</a></li>
</ul>

<p style="height: 30px"></p>
```

在本程序代码结构里，表示按钮组的 ul 元素语句中，其 class 属性值内含 round 字样，使得此按钮组具有**椭圆形**的风格。

图 5-84 所示的按钮组本身会占用浏览器窗口的**总宽度**，各个按钮**均分**其内部的宽度。此外，此按钮组的各个按钮应用了 Zurb's Foundation 的 secondary 颜色样式，因此其按钮的外观不同于前一个范例了。

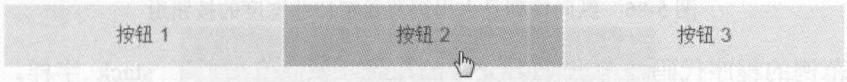


图 5-84 secondary 颜色样式的按钮组

```
<ul class="button-group even-3">
   <li><a href="#" class="button secondary">按钮 1</a></li>
   <li><a href="#" class="button secondary">按钮 2</a></li>
   <li><a href="#" class="button secondary">按钮 3</a></li>
</ul>
```

在本程序代码结构中，ul 元素的 class 属性值带有 even-3 字样，除了会使按钮组的宽度变成浏览器窗口的总宽度之外，还会使内部的各个按钮占用浏览器**窗口总宽度**的**三分之一**。

图 5-85 所示的是具有**椭圆形**样式的按钮组，它们占用了浏览器窗口的总宽度，一共有 6 个宽度相同的按钮。

图 5-85　椭圆样式且占用浏览器窗口总宽度的按钮组

```
<ul class="button-group round even-6">
   <li><a href="#" class="button">按钮 1</a></li>
   <li><a href="#" class="button">按钮 2</a></li>
   <li><a href="#" class="button">按钮 3</a></li>
   <li><a href="#" class="button">按钮 4</a></li>
   <li><a href="#" class="button">按钮 5</a></li>
   <li><a href="#" class="button">按钮 6</a></li>
</ul>

<p style="height: 30px"></p>
```

在上述程序代码结构中，ul 元素的 class 属性值带有 even-6 字样，除了会使按钮组的宽度变成浏览器窗口的总宽度之外，还会使内部的各个按钮占用浏览器**窗口总宽度**的**六分之一**。

图 5-86 所示的按钮组内含 3 个**纵向**排列且**都**占用浏览器窗口总宽度的按钮。

```
<ul class="stack button-group">
   <li><a href="#" class="button">按钮 1</a></li>
   <li><a href="#" class="button">按钮 2</a></li>
   <li><a href="#" class="button">按钮 3</a></li>
</ul>
<br>
```

图 5-86　纵向排列且占用浏览器窗口总宽度的按钮组

在上述范例的程序代码结构里，ul 元素的 class 属性值加上了 stack 字样，可使得内部的按钮以**纵向**来排列。

在图 5-87 中的椭圆形样式的按钮组里，内含 3 个宽度相同的按钮。

图 5-87　椭圆形样式的按钮组里内含 3 个宽度相同的按钮

当浏览器窗口宽度不足时，图 5-87 中的按钮组会转变成如图 5-88 所示的占用浏览器窗口宽度的**纵向**按钮组。

图 5-88 转变成占用浏览器窗口宽度的纵向按钮组

```
<ul class="stack-for-small round button-group">
  <li><a href="#" class="button">按钮 1</a></li>
  <li><a href="#" class="button">按钮 2</a></li>
  <li><a href="#" class="button">按钮 3</a></li>
</ul>

<p style="height: 30px"></p>
```

在上述范例的程序代码结构里，ul 元素的 class 属性值带有 stack-for-small 字样，可使得上述按钮组在**小型**窗口尺寸时转变成为**纵向**排列的**堆栈**（stack）型按钮组。

在图 5-89 中，有一个**工具栏**，内含 2 个**按钮组**，且按钮组之间有些**空隙**，而每个按钮组各含有 3 个按钮。

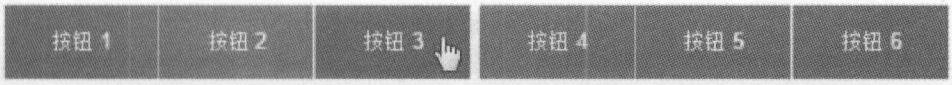


图 5-89 内含 2 个按钮组的工具栏

当浏览器窗口宽度不足时，图 5-89 中的工具栏会转变成为**纵向**排列的 2 个按钮组，如图 5-90 所示。

图 5-90 转变成为纵向排列的 2 个按钮组

```
<div class="button-bar">
  <ul class="button-group">
    <li><a href="#" class="small button">按钮 1</a></li>
    <li><a href="#" class="small button">按钮 2</a></li>
    <li><a href="#" class="small button">按钮 3</a></li>
  </ul>

  <ul class="button-group">
    <li><a href="#" class="small button">按钮 4</a></li>
    <li><a href="#" class="small button">按钮 5</a></li>
    <li><a href="#" class="small button">按钮 6</a></li>
```

```
    </ul>
  </div>

  <p style="height: 30px"></p>
```

在本范例的程序代码结构里，class 属性值为 button-bar 的 div 元素可成为一个**工具栏**；class 属性值为 button-group 的 ul 元素可成为一个**按钮组**；class 属性值为 small button 的 a 元素可成为一个**小型**尺寸的**按钮**。

在图 5-91 中的工具栏里，有一个**椭圆形**样式和一个**圆角矩形**样式的按钮组，而且分别应用了不同的**颜色**样式。

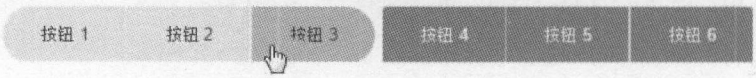


图 5-91　应用了不同颜色样式的椭圆按钮组和圆角矩形按钮组

当浏览器窗口宽度不足时，图 5-91 中的工具栏会转变成**纵向**排列的 2 个按钮组，如图 5-92 所示。

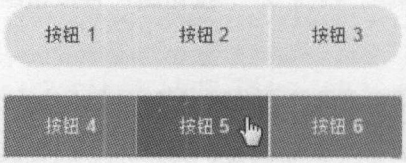


图 5-92　转变成纵向排列的 2 个按钮组

```
<!--行内混合样式的按钮组 -->
<div class="button-bar">
  <ul class="button-group round">
    <li><a href="#" class="small button secondary">按钮 1</a></li>
    <li><a href="#" class="small button secondary">按钮 2</a></li>
    <li><a href="#" class="small button secondary">按钮 3</a></li>
  </ul>

  <ul class="button-group radius">
    <li><a href="#" class="small button success">按钮 4</a></li>
    <li><a href="#" class="small button success">按钮 5</a></li>
    <li><a href="#" class="small button success">按钮 6</a></li>
  </ul>
</div>
```

上述按钮组主要由如下结构所共同构成。

- class 属性值为 button-group 的 ul 元素。其中 class 属性值可能加注如下字样。
  - radius，表示此按钮组的边框显示出椭圆形的模样。
  - round，表示此按钮组的边框显示出圆角矩形的模样。
  - even-3，表示此按钮组内的各个按钮显示窗口宽度的三分之一。

- even-6，表示此按钮组内的各个按钮显示窗口宽度的六分之一。
- stack，表示此按钮组内的各个按钮显示纵向排列的堆栈模样。
- stack-for-small，表示窗口宽度为小型尺寸的情况下，此按钮组内的各个按钮显示纵向排列的堆栈模样。
- success 或 secondary，表示此按钮组显示出 success 或secondary 颜色样式。
- small，表示此按钮组显示出小型尺寸的模样。

- class 属性值为 button-bar 的 div 元素。

在本范例的程序代码结构里，第一个按钮组所对应的 ul 元素的 class 属性值带有 round 字样，使得它具有**椭圆形**样式；而第二个按钮组所对应的 ul 元素的 class 属性值带有 radius 字样，使得它具有**圆角矩形**样式。

在第一个按钮组中的按钮所表示的 a 元素，其 class 属性值都带有 secondary 字样，即会应用 Zurb's Foundation 的 **secondary** 颜色样式；而在第二个按钮组中的按钮所表示的 a 元素，其 class 属性值都带有 success 字样，则会应用 **success** 颜色样式。

5.18 分割下拉式菜单按钮组

本节的程序代码都来自于 Ch05-18-split-buttons.html 范例文件。

在图 5-93 中的**分割下拉式菜单按钮组**的**右**侧带有**倒三角形**小图标的按钮可用来打开对应的菜单。

在此范例中，分割下拉式菜单按钮组的左侧按钮和右侧按钮可被设置不同的功能。当前右侧按钮是用来打开对应的菜单，而左侧按钮的对应功能在网页设计中则可通过 JavaScript 实现。

图 5-93 默认的分割下拉式菜单按钮

```
<a href="#" class="button split">默认的分割下拉式菜单按钮
   <span data-dropdown="drop1"></span></a><br>

<ul id="drop1" class="f-dropdown" data-dropdown-content>
   <li><a href="#">选项 1</a></li>
   <li><a href="#">选项 2</a></li>
   <li><a href="#">选项 3</a></li>
</ul>

<p></p>
```

该默认的分割下拉式菜单按钮主要由如下结构所共同构成。

- class 属性值为 button split 的 a 元素，并内含 data-dropdown 属性值为 drop1 的 span 元素。
- ul 元素，id 属性值为 drop1，class 属性值为 f-dropdown，而且带有无属性值的 data-dropdown-content 属性。

在本范例的程序代码结构中，设置 a 元素的 class 属性值为 button split，可使得 a 元素显示出**左侧**的按钮外观。再在 a 元素内部为 span 元素设置 data-dropdown 属性值为特定菜单的 ul 元素的 id 属性值（例如：drop1），即可显示出其**右侧**带有**倒三角形**小图标的按钮——用来打开特定菜单。

和前一个范例相比，图 5-94 中的分割下拉式菜单按钮组的**尺寸较小**，而且带有不同的颜色样式。

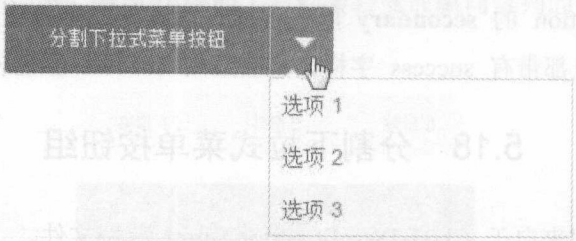


图 5-94　尺寸较小且颜色样式不同的分割下拉式菜单按钮

此分割下拉式菜单按钮主要由如下结构所共同构成。

- class 属性值为 small success radius button split 的 a 元素，并内含 data-dropdown 属性值为 drop2 的 span 元素。
- ul 元素，id 属性值为 drop2，class 属性值为 f-dropdown，而且带有无属性值的 data-dropdown-content 属性。

```
<a href="#" class="small success radius button split">分割下拉式菜单按钮
   <span data-dropdown="drop2"></span></a> <br>

<ul id="drop2" class="f-dropdown" data-dropdown-content>
   <li><a href="#">选项 1</a></li>
   <li><a href="#">选项 2</a></li>
   <li><a href="#">选项 3</a></li>
</ul>
```

在本范例的程序代码结构里，a 元素的 class 属性值加上 small success radius 字样之后，会使得分割下拉式菜单按钮组变成**小型**尺寸（small）、具有 success **颜色**样式和**圆角矩形**样式（radius）。

5.19 整体下拉式菜单按钮

本节的程序代码都来自于 Ch05-19-dropdown-buttons.html 范例文件。

在图 5-95 中的整体下拉式菜单按钮的范围之内，用鼠标单击，对应的菜单都会被打开。

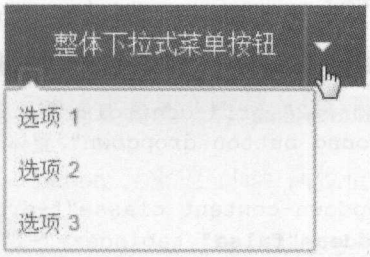


图 5-95 整体下拉式菜单按钮

```
    <button href="#" data-dropdown="drop1" aria-controls="drop1"
aria-expanded="false"
      class="button dropdown">整体下拉式菜单按钮</button><br>

    <ul id="drop1" data-dropdown-content class="f-dropdown" aria-hidden="true"
tabindex="-1">
       <li><a href="#">选项 1</a></li>
       <li><a href="#">选项 2</a></li>
       <li><a href="#">选项 3</a></li>
    </ul>

    <p></p>
```

此整体下拉式菜单按钮主要由如下结构所共同构成。

- button 元素，class 属性值为 button dropdown，data-dropdown 属性值和 aria-controls 属性值都为 drop1，aria-expanded 属性值为 false。
- ul 元素，id 属性值为 drop1，class 属性值为 f-dropdown，aria-hidden 属性值为 true，而且带有无属性值的 data-dropdown-content 属性。

在本范例的程序代码结构里，class 属性值为 button dropdown 的 button 元素里只有文字而不再有其他子元素，所以并没有显示出**左**、**右**分割的两个按钮。

而前两个范例，在 class 属性值为 button split 的 a 元素内还安排了一个 span 元素，并设置其 data-dropdown 属性值为特定菜单所对应的 ul 元素的 id 属性值（例如：drop1），因而才显示出用来打开特定菜单的**右侧**带有**倒三角形**的按钮。

和前一个范例进行比较，本范例（见图 5-96）的整体下拉式菜单按钮显示出**较大**尺寸的**椭圆形**，并且带有不同的**颜色样式**。

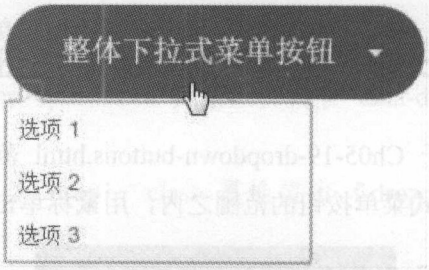


图 5-96　较大尺寸且带有不同颜色样式的整体下拉式菜单按钮

```
<button data-dropdown="drop2" aria-controls="drop2" aria-expanded="false"
   class="large alert round button dropdown">整体下拉式菜单按钮</button><br>

<ul id="drop2" data-dropdown-content class="f-dropdown"
    role="menu" aria-hidden="false" tabindex="-1">
   <li><a href="#">选项 1</a></li>
   <li><a href="#">选项 2</a></li>
   <li><a href="#">选项 3</a></li>
</ul>

<p></p>
```

此整体下拉式菜单按钮主要由如下元素所构成。

- button 元素，class 属性值为 large alert round button dropdown，data-dropdown 属性值和 aria-controls 属性值都为 drop2，aria-expanded 属性值为 false。
- ul 元素，id 属性值为 drop2，class 属性值为 f-dropdown，aria-hidden 属性值为 false，role 属性值为 menu，而且带有无属性值的 data-dropdown-content 属性。

在本范例的程序代码结构里，button 元素的 class 属性值带有 large alert round 字样，因此所显示出来的按钮具有**大型**尺寸（large）、**alert** 颜色样式和**椭圆形**（round）的外观。

在图 5-97 中是一个**被停用**的整体下拉式菜单按钮，其外观也表现得和一般按钮有所不同，鼠标指针移入其范围之内时不会转变成**手掌**形状，以此来告知用户此按钮当前是无法**操作**的。

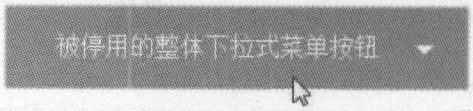


图 5-97　被停用的整体下拉式菜单按钮

```
<button data-dropdown="drop3" aria-controls="drop3" aria-expanded="false"
   class="disabled button dropdown">被停用的整体下拉式菜单按钮</button><br>

<ul id="drop3" data-dropdown-content class="f-dropdown" role="menu"
    aria-hidden="false" tabindex="-1">
   <li><a href="#">选项 1</a></li>
```

```
    <li><a href="#">选项 2</a></li>
    <li><a href="#">选项 3</a></li>
</ul>
```

这个被停用的整体下拉式菜单按钮主要由如下元素所构成。

- button 元素，class 属性值为 disabled button dropdown， data-dropdown 属性值和 aria-controls 属性值都为 drop3，aria-expanded 属性值为 false。
- ul 元素，id 属性值为 drop3，class 属性值为 f-dropdown，aria-hidden 属性值为 false，role 属性值为 menu，而且带有无属性值的 data-dropdown-content 属性。

在本范例的程序代码结构里，button 元素的 class 属性值加注了 disabled 字样，即可使得此按钮处于无法操作的**被停用**状态。

5.20 字体样式

本节的程序代码都来自于 Ch05-20-typography.html 范例文件。

在图 5-98 中，可以看到 6 行不同**字体**尺寸的标题文字。在 h1 至 h6 元素当中，设置其 class 属性值为 subheader或是安插内含文字的 small 元素，使得文字内容有被**刷淡**和**变小**的感觉。

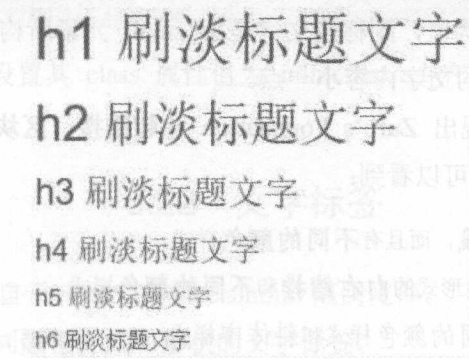


图 5-98 各种字体样式

```
<h1 class="subheader">h1 刷淡标题文字</h1>
<h2 class="subheader">h2 刷淡标题文字</h2>
<h3 class="subheader">h3 刷淡标题文字</h3>
<h4 class="subheader">h4 刷淡标题文字</h4>
<h5 class="subheader">h5 刷淡标题文字</h5>
<h6 class="subheader">h6 刷淡标题文字</h6>

<p style="height: 10px"></p>
```

在图 5-99 中，每个字体尺寸不同的标题文字里还分有字体尺寸**较大**的**主**标题文字和字体尺寸**较小**的**副**标题文字。

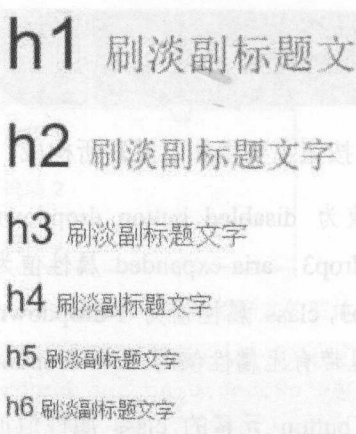


图 5-99　各种副标题字体样式

```
<h1>h1 <small>刷淡副标题文字</small></h1>
<h2>h2 <small>刷淡副标题文字</small></h2>
<h3>h3 <small>刷淡副标题文字</small></h3>
<h4>h4 <small>刷淡副标题文字</small></h4>
<h5>h5 <small>刷淡副标题文字</small></h5>
<h6>h6 <small>刷淡副标题文字</small></h6>

<p style="height: 10px"></p>
```

在本范例的程序代码结构里，副标题文字是由 small 元素所构成的。在**字体尺寸**上，会使得其文字内容比**上层元素**的文字内容**小**一点。

在图 5-100 中，特别表现出 Zurb's Foundation 的**超链接**、**区块引用**（block quote）和**一般引述**（cite）的默认风格，可以看到：

- **超链接**文字没有**下划线**，而且有**不同的颜色**样式。
- **区块引用**文字有着特定形式的**向右缩排**和**不同的颜色**样式。
- 一般引述文字有着不同的颜色样式和斜体字样式。

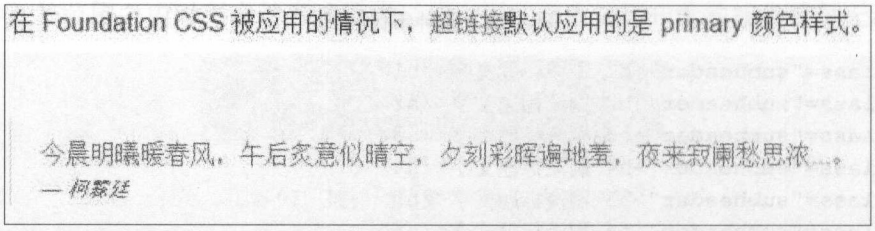


图 5-100　Zurb's Foundation 的超链接、区块引用和一般引述的默认风格

```
<p>在 Foundation CSS 被应用的情况下，<a href="#">超链接</a>
   默认被应用的是 primary 颜色样式。</p>

<p style="height: 10px"></p>
```

```
<blockquote>今晨明曦暖春风，午后炙意似晴空，夕刻彩晖遍地羞，夜来寂阑愁思浓....。
  <cite>柯霖廷</cite></blockquote>
```

在应用 Zurb's Foundation CSS的情况下，**a** 元素的**超链接**文字、**blockquote** 元素的**区块引用**文字和 **cite** 元素的**一般引述**文字都会出现外观上的变化。

5.21 行内列表

本节的程序代码都来自于 Ch05-21-inline-lists.html 范例文件。

在图 5-101 中的行内列表里，每个列表项都具有**横向排列**和**超链接特性**。

图 5-101 行内列表

```
<ul class="inline-list">
  <li><a href="#">选项 1</a></li>
  <li><a href="#">选项 2</a></li>
  <li><a href="#">选项 3</a></li>
  <li><a href="#">选项 4</a></li>
  <li><a href="#">选项 5</a></li>
</ul>
```

在 ul 元素中，主要设置其 class 属性值为 inline-list，并在内部每个 li 元素中再内含一个 a 元素。

5.22 文字标签

本节的程序代码都来自于 Ch05-22-labels.html 范例文件。

图 5-102 中有 4 个不同**颜色样式**和**形状**的文字标签。

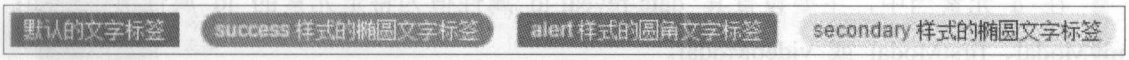


图 5-102 各种样式和形状的文字标签

```
<span class="label">默认的文字标签</span>  

<span class="success round label">success 样式的椭圆文字标签</span>  

<span class="alert radius label">alert 样式的圆角文字标签</span>  

<span class="secondary round radius label">secondary 样式的椭圆文字标签</span>

<p></p>
```

在 span 元素中，主要设置其 class 属性值为 label，并可考虑在其 class 属性值中：

- 加注 success、alert、secondary 等字样，使文字标签拥有不同的颜色样式。
- 加注 round 或 radius 字样，使文字标签拥有椭圆形或圆角矩形的边框样式。

在图 5-103 中，使用了 kbd 元素，内含特定文字内容，使得此文字内容具有文字标签的边框样式。

在 Windows 操作系统中，可按下 Windows 窗口键 + E 组合式快捷键，快速启动【资源管理器】窗口！

图 5-103　具有文字标签边框样式的文字内容

```
<p>在 Windows 操作系统中，可按下<kbd>Windows 窗口键</kbd>
   + <kbd>E</kbd>组合式快捷键，快速启动【资源管理器】窗口！ </p>
```

5.23　modal 窗格

本节的程序代码都来自于 Ch05-23-reveal-modal.html 范例文件。

所谓的 modal 窗格，是指可**动态**浮现在同一个网页页面的窗格。在图 5-104 中，其左侧是可打开 modal 窗格的超链接，其中间和右侧都是可打开 modal 窗格的按钮。

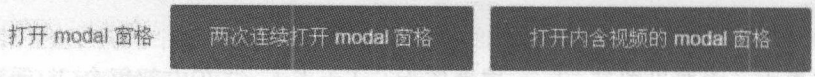


图 5-104　modal 窗格

```
<a href="#" data-reveal-id="myModal">打开 modal 窗格</a>  

<a href="#" data-reveal-id="firstModal" class="radius button">
   两次连续打开 modal 窗格</a>  

<a href="#" data-reveal-id="videoModal" class="radius button">
   打开内含视频的 modal 窗格</a>
```

在 a 元素当中，主要设置其 data-reveal-id 属性值为特定元素的 id 属性值，例如：myModal、firstModal 或 videoModal。

当单击图 5-105 所示的超链接时，会打开如图 5-106 所示的窗格。

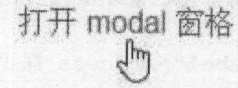


图 5-105　modal 窗格

若要关闭如上窗格，可在灰色区域单击一下，或是单击此窗格右上方的 ✕ 关闭按钮，即可回到页面原本的状态。

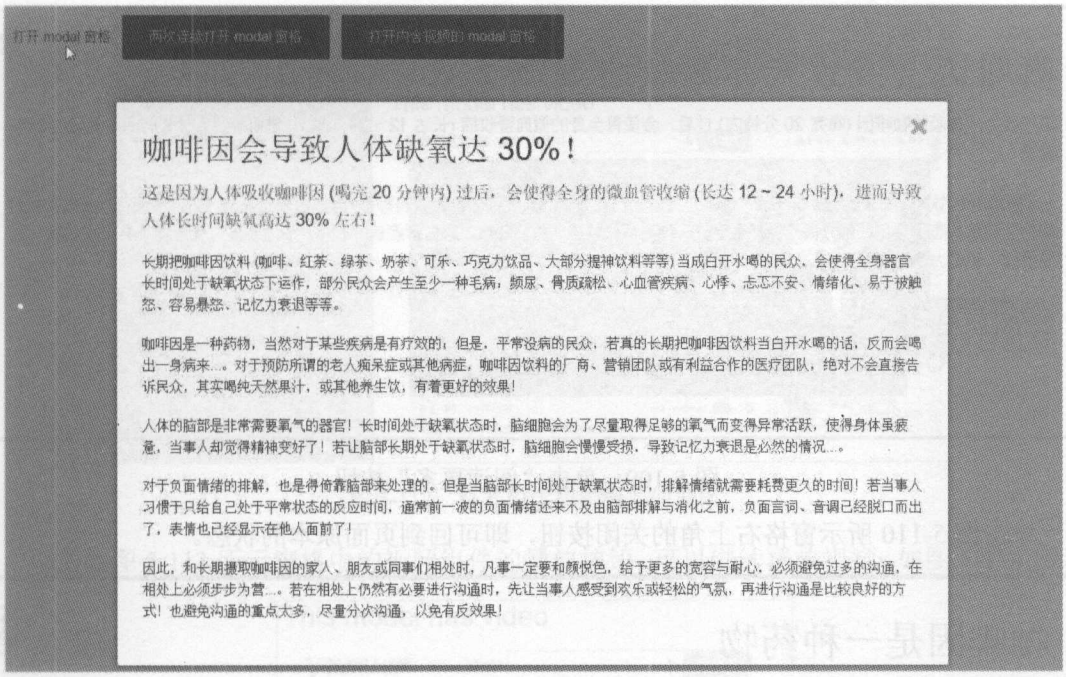


图 5-106　单击“打开 modal 窗格”的超链接后显示的窗格

单击如图 5-107 所示的按钮，可打开如图 5-108 所示的窗格。

图 5-107　单击“两次连续打开 modal 窗格”按钮

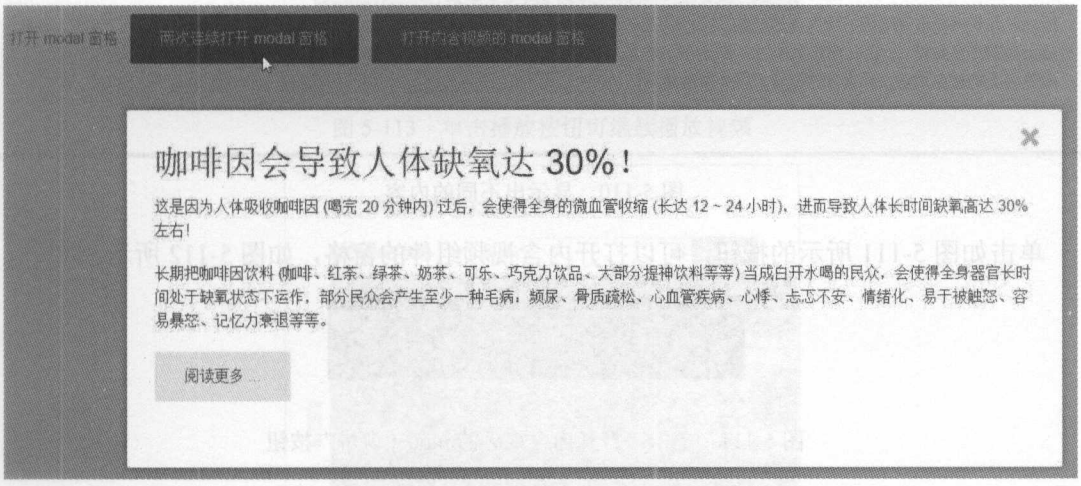


图 5-108　两次连续打开 modal 窗格

在如图 5-109 所示的窗格中单击【阅读更多】按钮，可以看到另一窗格出现在如上窗格的同一个位置，并内含如图 5-110 所示的不同内容。

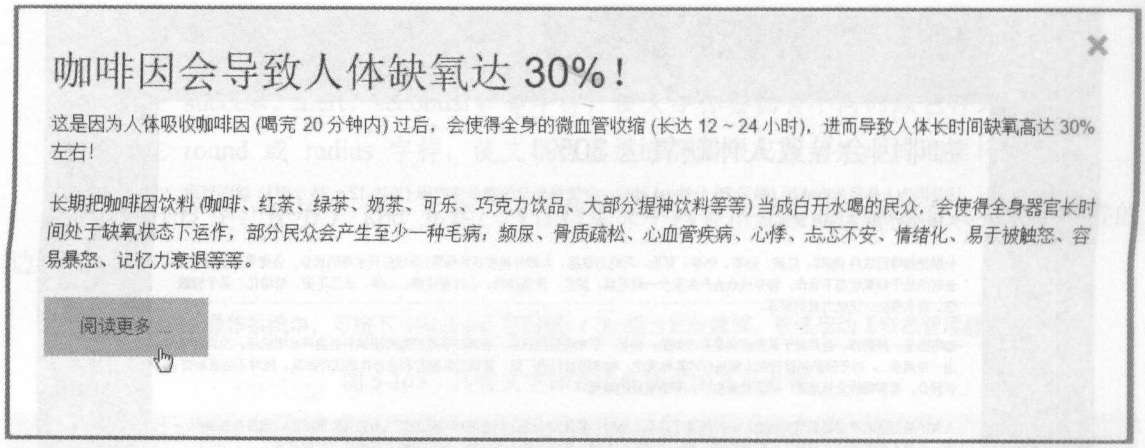


图 5-109　单击“阅读更多”按钮

单击图 5-110 所示窗格右上角的关闭按钮，即可回到页面原本的状态。

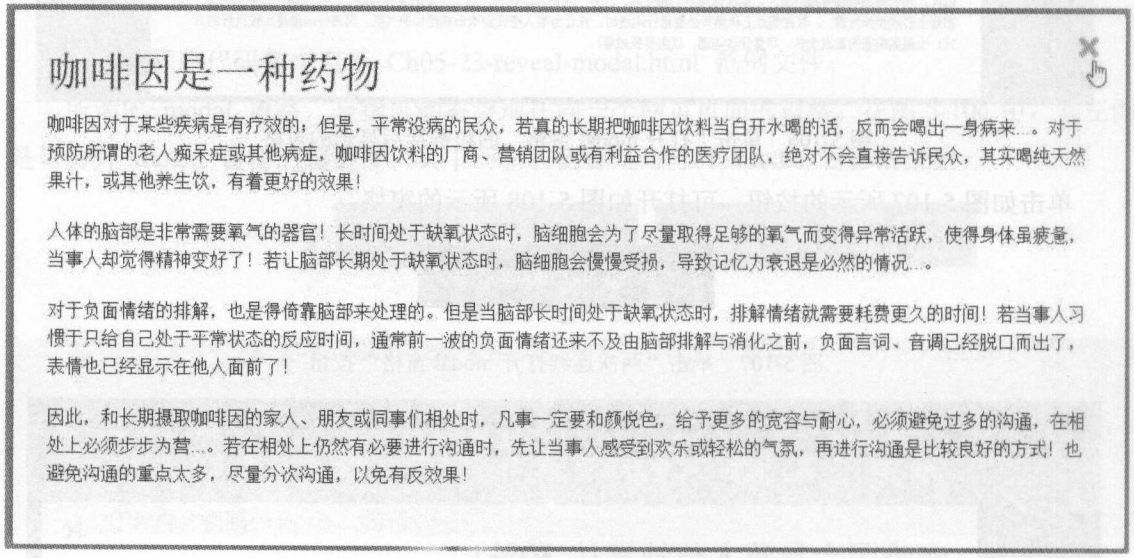


图 5-110　显示出不同的内容

单击如图 5-111 所示的按钮，可以打开内含视频组件的窗格，如图 5-112 所示。

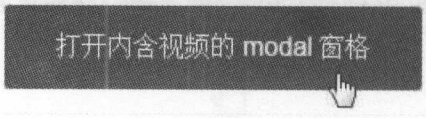


图 5-111　单击“打开内含视频的 modal 窗格”按钮

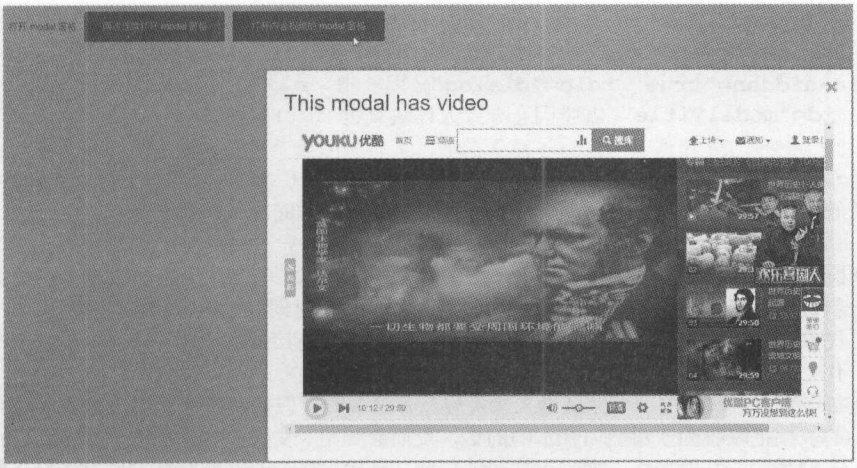


图 5-112　打开内含视频组件的窗格

单击如图 5-113 所示窗格中的视频组件的播放按钮，可以继续播放视频，如图 5-114 所示。

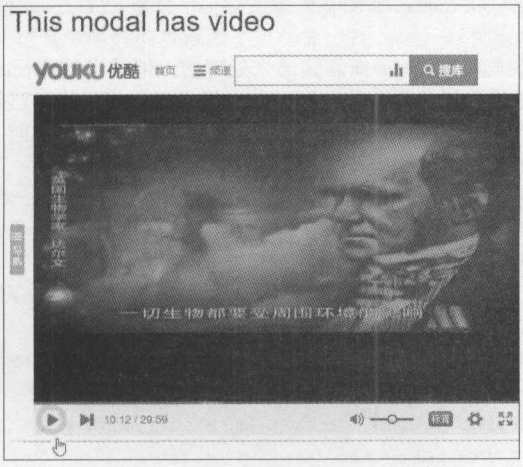


图 5-113　单击播放按钮可继续播放视频

图 5-114　继续播放视频

```
    <div id="myModal" class="reveal-modal" data-reveal
aria-labelledby="modalTitle"
      aria-hidden="true" role="dialog">
      <h2 id="modalTitle">咖啡因会导致人体缺氧达 30%! </h2>

      <p class="lead">这是因为人体吸收咖啡因（喝完 20 分钟内）过后，会使得全身的微血
         管收缩（长达 12 ~ 24 小时），进而导致人体长时间缺氧高达 30% 左右! </p>

      <p>长期把咖啡因饮料（咖啡、红茶、绿茶、奶茶、可乐、巧克力饮品、大部分提神饮
         料等等）当成白开水喝的民众，会使得全身器官长时间处于缺氧状态下运行，部分
         民众会产生至少一种毛病：频尿、骨质疏松、心血管疾病、心悸、忐忑不安、情绪
         化、易于被触怒、容易暴怒、记忆力衰退等等。</p>
      <p>咖啡因是一种药物，当然对于某些疾病是有疗效的；但是，平常没病的民众，若真
         的长期把咖啡因饮料当白开水喝的话，反而会喝出一身病来....。对于预防所谓的老
         人痴呆症或其他病症，咖啡因饮料的厂商、营销团队或有利益合作的医疗团队，绝
         对不会直接告诉民众，其实喝纯天然果汁，或其他养生饮，有着更好的效果! </p>
      <p>人体的脑部是非常需要氧气的器官! 长时间处于缺氧状态时，脑细胞会为了尽量取
         得足够的氧气而变得异常活跃，使得身体虽疲惫，当事人却觉得精神变好了! 若让
         脑部长期处于缺氧状态时，脑细胞会慢慢受损，导致记忆力衰退是必然的情况....
         </p>
      <p>对于负面情绪的排解，也是得倚靠脑部来处理的。但是当脑部长时间处于缺氧状态
         时，排解情绪就需要耗费更久的时间! 若当事人习惯于只给自己处于平常状态的反
         应时间，通常前一波的负面情绪还来不及由脑部排解与消化之前，负面言词、音调
         已经脱口而出了，表情也已经显示在他人面前了! </p>
      <p>因此，和长期摄取咖啡因的家人、朋友或同事们相处时，凡事一定要和颜悦色，给
         予更多的宽容与耐心，必须避免过多的沟通，在相处上必须步步为营....。若在相处
         上仍然有必要进行沟通时，先让当事人感受到欢乐或轻松的气氛，再进行沟通是比
         较良好的方式! 也避免沟通的重点太多，尽量分次沟通，以免有反效果! </p>

      <a class="close-reveal-modal" aria-label="Close">&#215;</a>
    </div>

    <!-- Reveal Modals begin -->
    <div id="firstModal" class="reveal-modal" data-reveal
aria-labelledby="firstModalTitle"
      aria-hidden="true" role="dialog">
      <h2 id="firstModalTitle">咖啡因会导致人体缺氧达 30%! </h2>

      <p>这是因为人体吸收咖啡因（喝完 20 分钟内）过后，会使得全身的微血管收缩（长
         达 12~ 24 小时），进而导致人体长时间缺氧高达 30% 左右! </p>
      <p>长期把咖啡因饮料（咖啡、红茶、绿茶、奶茶、可乐、巧克力饮品、大部分提神饮
         料等等）当成白开水喝的民众，会使得全身器官长时间处于缺氧状态下运行，部分
         民众会产生至少一种毛病：频尿、骨质疏松、心血管疾病、心悸、忐忑不安、情绪
         化、易于被触怒、容易暴怒、记忆力衰退等等。</p>
      <p><a href="#" data-reveal-id="secondModal" class="secondary button">
         阅读更多 ...</a></p>

      <a class="close-reveal-modal" aria-label="Close">&#215;</a>
    </div>
```

```
<div id="secondModal" class="reveal-modal" data-reveal
   aria-labelledby="secondModalTitle" aria-hidden="true" role="dialog">
   <h2 id="secondModalTitle">咖啡因是一种药物</h2>

   <p>咖啡因对于某些疾病是有疗效的；但是，平常没病的民众，若真的长期把咖啡因饮
      料当白开水喝的话，反而会喝出一身病来....。对于预防所谓的老年性痴呆或其他病
      症，咖啡因饮料的厂商、营销团队或有利益合作的医疗团队，绝对不会直接告诉民
      众，其实喝纯天然果汁，或其他养生饮，有着更好的效果！</p>
   <p>人体的脑部是非常需要氧气的器官！长时间处于缺氧状态时，脑细胞会为了尽量取
      得足够的氧气而变得异常活跃，使得身体虽疲惫，当事人却觉得精神变好了！若让
      脑部长期处于缺氧状态时，脑细胞会慢慢受损，导致记忆力衰退是必然的情况....。
      </p>
   <p>对于负面情绪的排解，也是得倚靠脑部来处理的。但是当脑部长时间处于缺氧状态
      时，排解情绪就需要耗费更久的时间！若当事人习惯于只给自己处于平常状态的反
      应时间，通常前一波的负面情绪还来不及由脑部排解与消化之前，负面言词、音调
      已经脱口而出了，表情也已经显示在他人面前了！</p>
   <p>因此，和长期摄取咖啡因的家人、朋友或同事们相处时，凡事一定要和颜悦色，给
      予更多的宽容与耐心，必须避免过多的沟通，在相处上必须步步为营....。若在相处
      上仍然有必要进行沟通时，先让当事人感受到欢乐或轻松的气氛，再进行沟通是比
      较良好的方式！也避免沟通的重点太多，尽量分次沟通，以免有反效果！</p>

   <a class="close-reveal-modal" aria-label="Close">&#215;</a>
</div>

<div id="videoModal" class="reveal-modal large" data-reveal
    aria-labelledby="videoModalTitle" aria-hidden="true" role="dialog">

   <h2 id="videoModalTitle">This modal has video</h2>

   <div class="flex-video widescreen vimeo">
     <iframe width="1280" height="720"
       src=" http://v.youku.com/v_show/id_XNzA0ODYyMTY=.html?f=5517394"
       frameborder="0" allowfullscreen></iframe>
   </div>

   <a class="close-reveal-modal" aria-label="Close">&#215;</a>

</div>
```

上述窗格再次通过如下方式之一制作出可调用的modal窗格组件。

- class 属性值为 reveal-modal、aria-labelledby 属性值为 modalTitle、aria-hidden 属性值为 true、role 属性值为 dialog 而且带有无属性值的 data-reaveal 属性的 div 元素。
- class 属性值为 reveal-modal、aria-labelledby 属性值为 firstModalTitle、aria-hidden 属性值为 true、role 属性值为 dialog 而且带有无属性值的 data-reaveal 属性的 div 元素。
- class 属性值为 reveal-modal、aria-labelledby 属性值为 secondModalTitle、aria-hidden 属性值为 true、role 属性值为 dialog 而且带有无属性值的 data-reaveal 属性的 div 元素。

- class 属性值为 reveal-modal large、aria-labelledby 属性值为 videoModalTitle、role 属性值为 dialog 而且带有无属性值的 data-reaveal 属性的 div 元素。

最后，再通过如下方式提供关闭 modal 窗格的组件。

在 a 元素中，设置其 class 属性值为 close-reveal-modal、aria-label 属性值为 Close。

本范例的程序代码结构相当复杂，若对于 HTML 和 CSS 语言的语句不够熟悉，则可能会遇到理解上的极大阻碍。请读者要特别注意各个被打开的 modal 窗格的内容，它们对应到哪部分的语句，这样可以大致自行解析其程序代码的意义。

5.24 警告框

本节的程序代码都来自于 Ch05-24-alerts.html 范例文件。

在图 5-115 中有 5 个不同样式的警告框，都内含一小段文字和**右侧**可隐藏本身的**关闭按钮**。

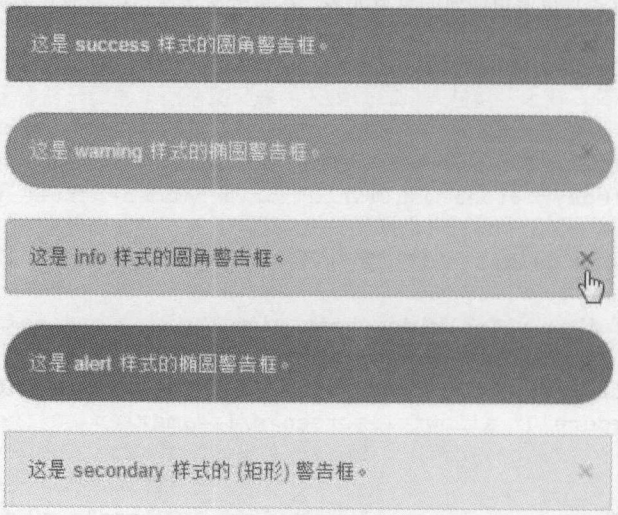


图 5-115 各种样式的警告框

在 div 元素中，设置其 class 属性值为 alert-box，并可在 class 属性值中加注：

- success、warning、info、secondary 等，以改变此警告框的背景颜色。
- radius 或 round，以使得此警告框具有圆角矩形或椭圆形的边框样式。

并可在上述的 div 元素当中安排一个 class 属性值为 close 的 a 元素，为用户提供可动态关闭其警告框的关闭按钮。

在图 5-116 中，警告框带有 **success 颜色样式**和圆角矩形样式，右侧的关闭按钮可隐藏此警告框。

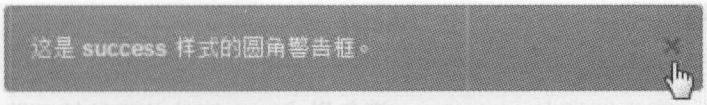


图 5-116 带有 success 颜色样式和圆角矩形样式的警告框

```
<div data-alert class="alert-box success radius">
   这是 success 样式的圆角警告框。
   <a href="#" class="close">&times;</a>
</div>
```

在本范例的程序代码结构里，由 × 语句构成了**警告框**右侧的**关闭按钮**。

在图 5-117 中，警告框带有 **warning** 颜色样式和**椭圆形**样式，右侧的关闭按钮可隐藏此警告框。

图 5-117 带有 warning 颜色样式和椭圆形样式的警告框

```
<div data-alert class="alert-box warning round">
   这是 warning 样式的椭圆警告框。
   <a href="#" class="close">&times;</a>
</div>
```

在图 5-118 中，警告框带有 **info** 颜色样式和**圆角矩形**样式，右侧的关闭按钮可隐藏此警告框。

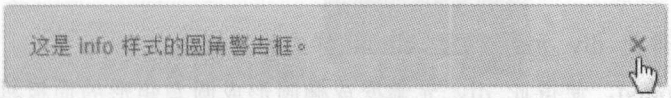


图 5-118 带有 info 颜色样式和圆角矩形样式的警告框

```
<div data-alert class="alert-box info radius">
   这是 info 样式的圆角警告框。
   <a href="#" class="close">&times;</a>
</div>
```

在图 5-119 中，警告框带有 **alert** 颜色样式和椭圆形样式，右侧的关闭按钮可隐藏此警告框。

图 5-119 带有 alert 颜色样式和椭圆形样式的警告框

```
<div data-alert class="alert-box alert round">
   这是 alert 样式的椭圆警告框。
```

```
    <a href="#" class="close">&times;</a>
</div>
```

在图 5-120 中，警告框带有 secondary **颜色样式**，右侧的关闭按钮可隐藏此警告框。

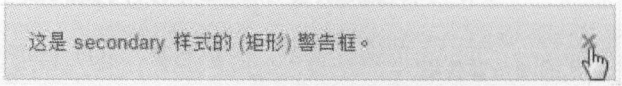


图 5-120　带有 secondary 颜色样式的警告框

```
<div data-alert class="alert-box secondary">
    这是 secondary 样式的 (矩形) 警告框。
    <a href="#" class="close">&times;</a>
</div>
```

5.25　面板框

本节的程序代码都来自于 Ch05-25-panels.html 范例文件。

在图 5-121 中有一个默认样式的面板框，内含**标题**和**文字内容**。

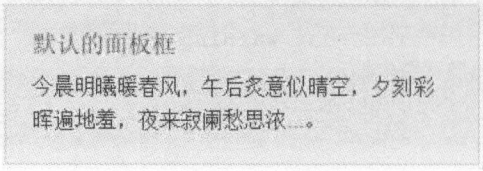


图 5-121　默认样式的面板框

在 div 元素中，设置其 class 属性值为 panel，并可在 class 属性值中加注：

- callout，使得此 div 元素变成 callout 样式的面板框。
- radius 或 round，使得此 div 元素变成椭圆形或圆角矩形的面板框。

```
<div class="panel">
    <h5>默认的面板框</h5>

    <p>今晨明曦暖春风，午后炙意似晴空，夕刻彩晖遍地羞，夜来寂阑愁思浓....。</p>
</div>

<p></p>
```

在图 5-122 中，有一个 callout 样式的面板框，内含标题和文字内容。

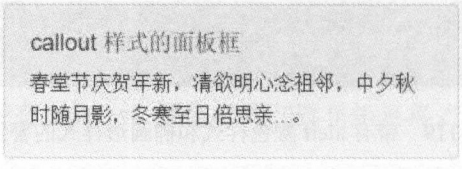


图 5-122　callout 样式的面板框

```
<div class="panel callout radius">
   <h5>callout 样式的面板框</h5>

   <p>春堂节庆贺年新，清欲明心念祖邻，中夕秋时随月影，冬寒至日倍思亲....。</p>
</div>
```

5.26 动态提示框

本节的程序代码都来自于 Ch05-26-tooltips.html 范例文件。

当鼠标指针移到**左**侧文字时，会出现一个在**下方**的**矩形**动态提示框，如图 5-123 所示。

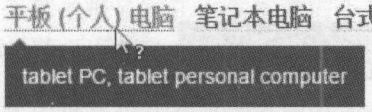


图 5-123 显示在下方的矩形动态提示框

```
<span data-tooltip aria-haspopup="true" class="has-tip"
   title="tablet PC, tablet personal computer">
   平板 (个人) 计算机
</span>  
```

在本范例的程序代码结构里，动态提示框内的文字是 span 元素的 title 属性值。再对此 span 元素设置无属性值的 data-tooltip 属性，并设置 aria-haspopup 属性值为 true 与 class 属性值为 has-tip，以使得此动态提示框有特别的外观。

当鼠标指针移到**中间文字**时，会出现一个在**上方**的**圆角矩形**动态提示框，如图 5-124 所示。

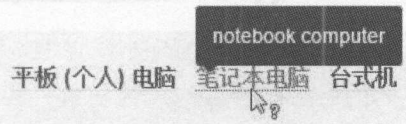


图 5-124 显示在上方的圆角矩形动态提示框

```
<span data-tooltip aria-haspopup="true" class="has-tip tip-top radius"
   title="notebook computer">
   笔记本电脑
</span>  
```

在本范例的程序代码结构里，tip-top 可使得动态提示框显示在**中间**文字的**上方**，而 radius 会使得动态提示框显示出**圆角矩形**。

当鼠标指针移到右侧文字时，会出现一个在**下方**的**椭圆形**动态提示框，如图 5-125 所示。

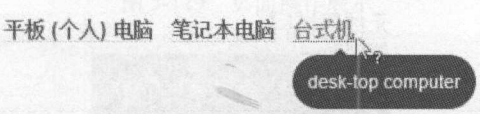


图 5-125 显示在下方的椭圆形动态提示框

```
<span data-tooltip aria-haspopup="true" class="has-tip tip-bottom round"
   title="desk-top computer">
   台式机
</span>

<span style="display: inline-block ; width: 250px"> </span>
```

在本范例的程序代码结构里，tip-bottom 可使得动态提示框显示在**右侧**文字的**下方**，而 round 会使得动态提示框显示出**椭圆形**。

当鼠标指针移到图 5-126 所示的**左**侧文字上时，会出现一个在**左方**的**圆角矩形**动态提示框。

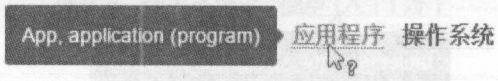


图 5-126　显示在左边的圆角矩形动态提示框

```
<span data-tooltip aria-haspopup="true" class="has-tip tip-left radius"
    title="App, application (program)">
   应用程序
</span>  
```

在本范例的程序代码结构里，tip-left 可使得动态提示框显示在**左侧**文字的**左方**，而 radius 会使得动态提示框显示出**圆角矩形**。

当鼠标指针移到图 5-127 所示的**右**侧文字上时，会出现一个在**右方**的**椭圆形**动态提示框。

图 5-127　显示在右边的椭圆形动态提示框

```
<span data-tooltip aria-haspopup="true" class="has-tip tip-right round"
    title="OS, operating system">
   操作系统
</span>

<span style="display: inline-block ; width: 250px"> </span>
```

在本范例的程序代码结构里，tip-right 可使得动态提示框显示在**右侧**文字的**右方**，而 round 会使得动态提示框显示出**椭圆形**。

当鼠标指针移到图 5-128 所示的文字上时，会出现一个在**下方**的动态提示框。但是当用户在**触控**设备上将鼠标指针移入此文字范围内时并不会显示下方的动态提示框。

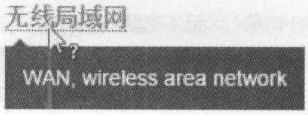


图 5-128　显示在下方的动态提示框

```
    <span data-tooltip aria-haspopup="true" data-options="disable_for_touch:
true"
          class="has-tip" title="WAN, wireless area network">
       无线局域网络
    </span>
```

动态提示框的实现是在 span 元素中设置 class 属性值为 has-tip、aria-haspopup 属性值为 true、title 属性值为特定文字（例如：tablet PC, tablet personal computer），而且带有无属性值的 data-tooltip 属性。其中，其 class 属性值可加注如下字样。

- tip-top，表示此动态提示框显示在元素本身的上方。
- tip-bottom，表示此动态提示框显示在元素本身的下方。
- tip-left，表示此动态提示框显示在元素本身的左侧。
- tip-right，表示此动态提示框显示在元素本身的右侧。
- radius 或 round，表示此动态提示框以椭圆形或圆角矩形来显示。

在本范例的程序代码结构中，受 data-options 属性值为 disable_for_touch: true 的影响，其对应的动态提示框在**触控**设备的屏幕上将不会被显示出来。

5.27 页面操作展示

本节的程序代码都来自于 Ch05-27-joyride.html 范例文件。

在图 5-129 中的页面操作展示里，可以看到网页本来的静态页面变成**灰色**模样，而操作展示的对话框则被清晰地浮现在页面之上。

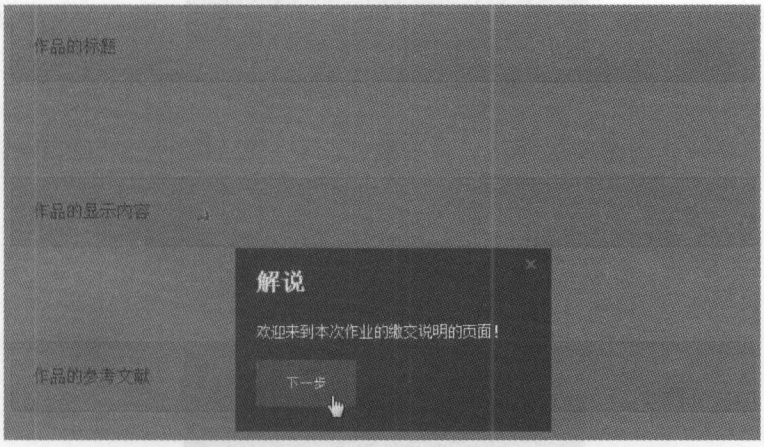


图 5-129 页面操作展示

```
    <div id="firstStop" class="panel">作品的标题</div>

    <p style="height: 50px"></p>
```

```
<div id="numero1" class="panel">作品的显示内容</div>

<p style="height: 50px"></p>

<div id="numero2" class="panel">作品的参考文献</div>
```

笔者通过如上的程序代码，让浏览器搭建此网页的基本内容，以让读者能清楚地看到基本内容变成**较深灰色**区域之后，才会出现**页面操作展示**的**解说**，如图 5-130 所示。

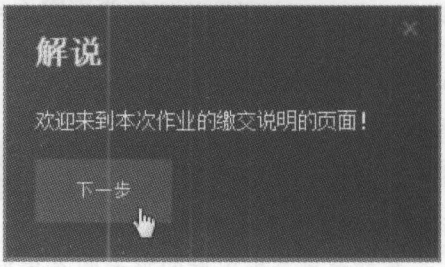


图 5-130　页面操作展示的解说

单击图 5-130 中的【下一步】按钮，可以看到如图 5-131 所示的**第二个**窗格被打开。

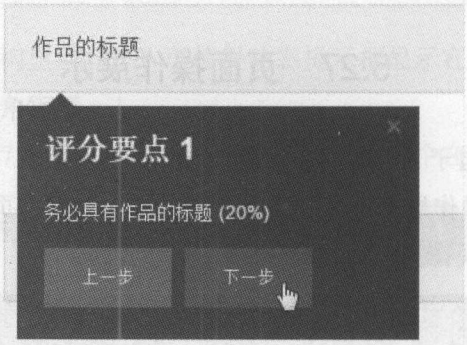


图 5-131　第二个窗格被打开

在图 5-131 中单击窗格中的【下一步】按钮，可以看到如图 5-132 所示的**第三个**窗格被打开。

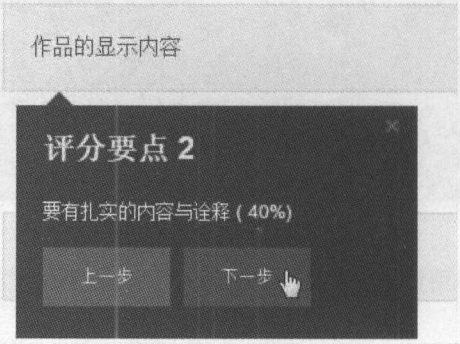


图 5-132　第三个窗格被打开

在图 5-132 中单击窗格中的【下一步】按钮，可以看到如图 5-133 所示的**第四个**窗格被打开。

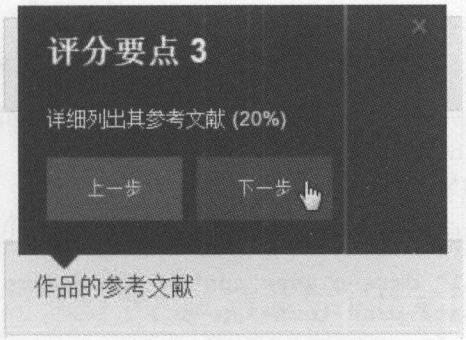


图 5-133　第四个窗格被打开

在图 5-133 中单击窗格中的【下一步】按钮，可以看到如图 5-134 所示的**第五个**窗格被打开。

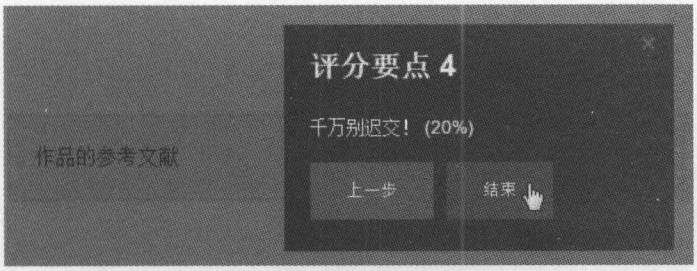


图 5-134　第五个窗格被打开

在图 5-134 中单击窗格中的【结束】按钮，可以看到页面回到原本的状态，如图 5-135 所示。

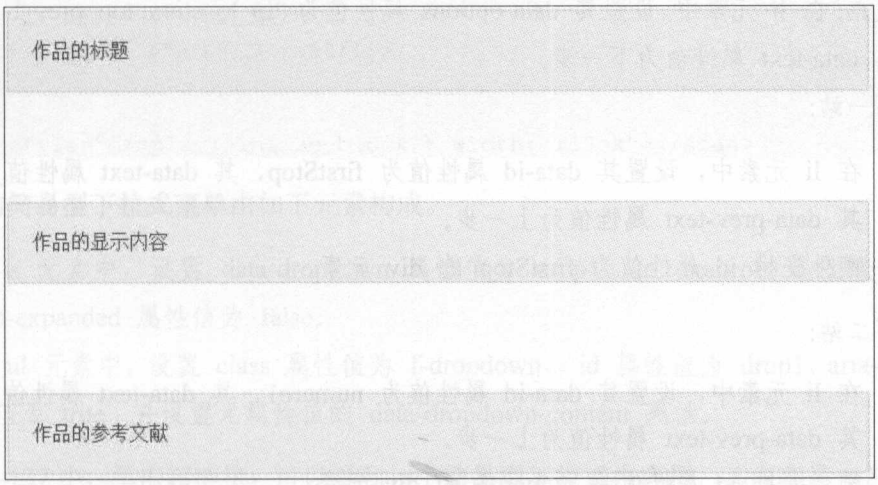


图 5-135　页面回到原本的状态

```
    <ol class="joyride-list" data-joyride>
      <li data-text="下一步" data-options="tip_location: top; prev_button:
false">
        <h4>解说</h4><br>
        <p>欢迎来到本次作业的缴交说明的页面！</p>
      </li>

      <li data-id="firstStop" data-text="下一步" data-prev-text="上一步">
        <h4>评分要点 1</h4><br>
        <p>务必具有作品的标题 (20%)</p>
      </li>

    <li data-id="numero1" data-class="custom so-awesome"
    data-text="下一步" data-prev-text="上一步">
        <h4>评分要点 2</h4><br>
        <p>要有扎实的内容与诠释 ( 40%) </p>
      </li>

    <li data-id="numero2" data-button="下一步" data-prev-text="上一步"
          data-options="tip_location:top; tip_animation:fade">
        <h4>评分要点 3</h4><br>
        <p>详细列出其参考文献 (20%)</p>
      </li>

      <li data-button="结束" data-prev-text="上一步">
        <h4>评分要点 4</h4><br>
        <p>千万别迟交！ (20%)</p>
      </li>
    </ol>
```

在 ol 元素中，设置其 class 属性值为 joyride-list，并设置无属性值的 data-joyride 属性。在页面操作展示中，如下设置各站元素。

- 起点：在 li 元素中，设置其 data-options 属性值为 tip_location: top; prev_button: false、其 data-text 属性值为下一步。
- 第一站：
  - 在 li 元素中，设置其 data-id 属性值为 firstStop、其 data-text 属性值为下一步、其 data-prev-text 属性值为上一步。
  - 额外安排 id 属性值为 firstStop 的 div 元素。
- 第二站：
  - 在 li 元素中，设置其 data-id 属性值为 numero1、其 data-text 属性值为下一步、其 data-prev-text 属性值为上一步。
  - 额外安排 id 属性值为 numero1 的 div 元素。

- 第三站：
  - 在 li 元素中，设置其 data-id 属性值为 numero2、其 data-button 属性值为下一步、其 data-prev-text 属性值为上一步、其 data-options 属性值为 tip_location:top; tip_animation: fade。
  - 额外安排 id 属性值为 numero2 的 div 元素。
- 最后一站：

  在 li 元素中，设置其 data-button 属性值为结束、其 data-pre-text为上一步。

5.28 简易型下拉式菜单与下拉式内容框

本节的程序代码都来自于 Ch05-28-dropdowns.html 范例文件。

通过单击图 5-136 中的超链接，会出现对应的简易型下拉式菜单，使整个菜单外观看起来相当简单（可以看到有 3 个选项）。

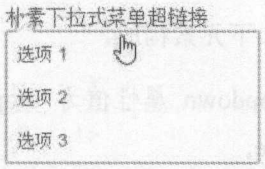


图 5-136 简易型下拉式菜单

```
<a data-dropdown="drop1" aria-controls="drop" aria-expanded="false">
   朴素下拉式菜单超链接</a>

<ul id="drop1" class="f-dropdown" data-dropdown-content aria-hidden="true"
    tabindex="-1">
   <li><a href="#">选项 1</a></li>
   <li><a href="#">选项 2</a></li>
   <li><a href="#">选项 3</a></li>
</ul>

<span style="display: inline-block ; width: 150px"></span>
```

上述的**简易型下拉式菜单**由如下元素构成。

- 在 a 元素中，设置 data-dropdown 属性值为 drop1、aria-controls 属性值为 drop、aria-expanded 属性值为 false。
- 在 ul 元素中，设置 class 属性值为 f-dropdown、id 属性值为 drop1、aria-hidden 属性值为 true，并设置无属性值的 data-dropdown-content 属性。

在图 5-137 中，单击超链接，可以看到一个**简易型下拉式内容框**，其内容是一首诗，而不是菜单的选项。

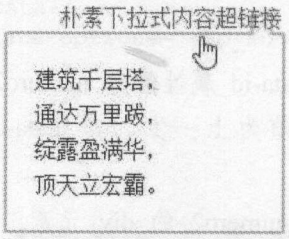


图 5-137　简易型下拉式内容框

```
<a data-dropdown="drop2" aria-controls="drop" aria-expanded="false">
   朴素下拉式内容超链接</a>

<div id="drop2" data-dropdown-content class="f-dropdown content"
    aria-hidden="true" tabindex="-1">
   <p> 建筑千层塔，<br>通达万里跋，<br>
       绽露盈满华，<br>顶天立宏霸。</p>
</div>

<p style="height: 100px"></p>
```

上述的**简易型下拉式内容框**由如下元素构成。

- 在 a 元素中，设置 data-dropdown 属性值为 drop2、 aria-controls 属性值为 drop、aria-expanded 属性值为 false。
- 在 div 元素中，设置 class 属性值为 f-dropdown content、 id 属性值为 drop2、aria-hidden 属性值为 true，并设置无属性值的 data-dropdown-content 属性。

在图 5-138 中，单击**菜单按钮**之后，会看到另一种**简易型下拉式菜单**，内含 3 个选项。

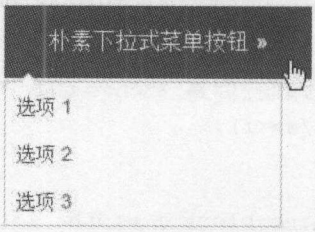


图 5-138　另一种样式的简易型下拉式菜单

```
<a href="#" class="button" data-dropdown="drop3">
   朴素下拉式菜单按钮&raquo;</a>

<ul id="drop3" class="f-dropdown" data-dropdown-content>
   <li><a href="#">选项 1</a></li>
   <li><a href="#">选项 2</a></li>
   <li><a href="#">选项 3</a></li>
</ul>

<p style="height: 100px"></p>
```

上述下拉式菜单组件由如下元素构成。

- 在 a 元素中，设置 data-dropdown 属性值为 drop3、 class 属性值为 button。
- 在 ul 元素中，设置 class 属性值为 f-dropdown、 id 属性值为 drop3，并设置无属性值的 data-dropdown-content 属性。

在图 5-139 中，单击**菜单按钮**之后，会在其**右侧**出现另一种**简易型下拉式菜单**，内含 3 个选项。

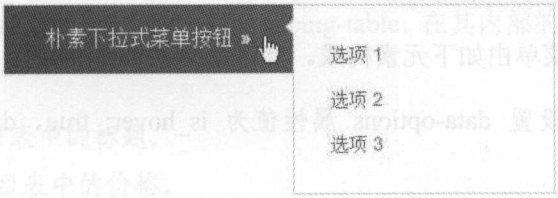


图 5-139 在右侧显示的简易型下拉式菜单

```
<a href="#" data-options="align:right" data-dropdown="drop4" class="button">
   朴素下拉式菜单按钮&raquo;
</a>

<ul id="drop4" class="content f-dropdown" data-dropdown-content>
   <li><a href="#">选项 1</a></li>
   <li><a href="#">选项 2</a></li>
   <li><a href="#">选项 3</a></li>
</ul>

<p style="height: 100px"></p>
```

上述简易型下拉式菜单由如下元素构成。

- 在 a 元素中，设置 data-options 属性值为 align:right、 data-dropdown 属性值为 drop4、 class 属性值为 button。
- 在 ul 元素中，设置 class 属性值为 content f-dropdown、id 属性值为 drop4，并设置无属性值的 data-dropdown-content 属性。

在图 5-140 中，将鼠标指针**移入**超链接，即可打开简易型下拉式菜单；当鼠标指针**移出**时，其菜单便会自动消失。

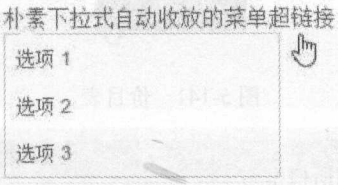


图 5-140 将鼠标指针移入超链接即可打开的简易型下拉式菜单

```
<a href="#" data-dropdown="hover1" data-options="is_hover:true">
   朴素下拉式自动收放的菜单超链接
</a>

<ul id="hover1" class="f-dropdown" data-dropdown-content>
   <li><a href="#">选项 1</a></li>
   <li><a href="#">选项 2</a></li>
   <li><a href="#">选项 3</a></li>
</ul>

<p style="height: 100px"></p>
```

上述简易型下拉式菜单由如下元素构成。

- 在 a 元素中，设置 data-options 属性值为 is_hover: true、data-dropdown 属性值为 hover1。
- 在 ul 元素中，设置 class 属性值为 f-dropdown、 id 属性值为 hover1，并设置无属性值的 data-dropdown-content 属性。

5.29 价目表

本节的程序代码都来自于 Ch05-29-pricing-tables.html 范例文件。

在图 5-141 中的价目表里，可以看到有特定的外观样式，除了每**行**只有单个单元格之外，各个单元格内的文字都为**居中对齐**。

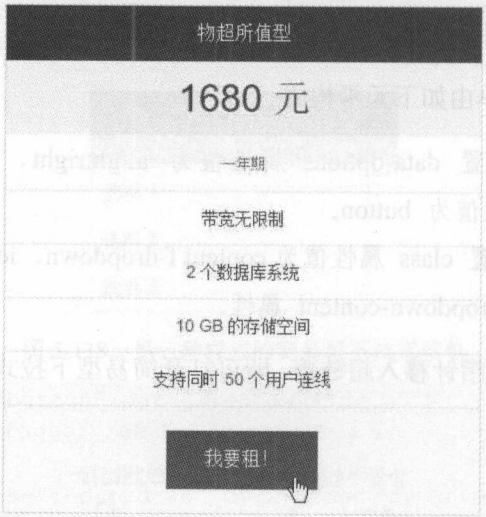


图 5-141 价目表

```
<ul class="pricing-table">
   <li class="title">物超所值型</li>
   <li class="price">1680 元</li>
```

```
    <li class="description">一年期</li>
    <li class="bullet-item">带宽无限制</li>
    <li class="bullet-item">2 个数据库系统</li>
    <li class="bullet-item">10 GB 的存储空间</li>
    <li class="bullet-item">支持同时 50 个用户连线</li>

    <li class="cta-button"><a class="button" href="#">我要租! </a></li>
</ul>

<p style="height: 100px"></p>
```

在 ul 元素中，设置其 class 属性值为 pricing-table。在其内部的 li 元素中，设置其 class 属性值分别如下。

- title，表示价目表中的标题。
- price，表示价目表中的价格。
- description，表示价目表中的说明。
- bullet-item，表示价目表中的特定项。
- cta-button，表示价目表中的立即购物按钮。

5.30 进度条

本节的程序代码都来自于 Ch05-30-progress-bar.html 范例文件。

在图 5-142 中，有 3 个**进度条**，可用来表达某一事件的**当前进度**。图 5-143 到图 5-145 就是这 3 个分开的进度条。

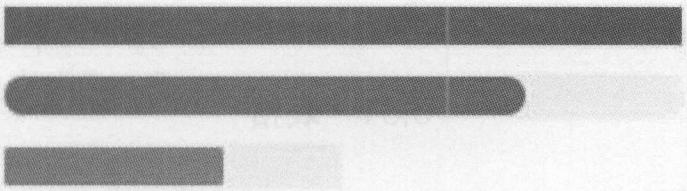


图 5-142 3 个不同样式的进度条

图 5-143 蓝色矩形样式的进度条

```
<div class="progress">
    <span class="meter"></span>
</div>
```

图 5-144 红色椭圆形样式的进度条

```
<div class="progress large-9 alert round">
  <span class="meter" style="width: 77%"></span>
</div>
```

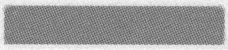


图 5-145　绿色圆角矩形样式的进度条

```
<div class="progress small-6 success radius">
  <span class="meter" style="width: 66%"></span>
</div>
```

上述进度条中 span 元素的 style 属性值可设置如下。

- width: 77%，可使得进度条总宽度的 77% 成为**当前进度长条**的宽度，以便让用户知道**当前的进度**。
- width: 66%，可使得进度条总宽度的 66% 成为**当前进度长条**的宽度，以便让用户知道**当前的进度**。

在 div 元素中，设置 class 属性值为 progress，并在内部安排 class 属性值为 meter的 span 元素。其中，div 元素的 class 属性值可加注如下字样。

- large-9，表示处于大尺寸窗口宽度时，此进度条显示 12 等分当中 9 等分的宽度，也就是**四分之三**（ 9 / 12）的窗口宽度。
- small-6，表示处于小尺寸窗口宽度时，此进度条显示 12 等分当中 6 等分的宽度，也就是**二分之一**（6 / 12）的窗口宽度。
- alert、success 等，使进度条显示出 alert 或 success 颜色样式。
- round 或 radius，使进度条的边框显示出椭圆形或圆角矩形的样式。

5.31　表格

本节的程序代码都来自于 Ch05-31-tables.html 范例文件。

在图 5-146 中，有一个 4 行 4 列、带有表格标题（史上最贵水果）和列标题（水果名称、原产地等）的表格。

史上最贵水果

水果名称	原产地	库存量 (斤)	价格 (每斤)
有机磁力火龙果	水星	348	100 万元
有机离散榴莲	火星	850	200 万元
有机共轭山竹果	飞马座星群	1516	500 万元

图 5-146　4×4 的表格

应用 Zurb's Foundation CSS之后，此网页中的 **table** 元素及各个子元素的外观会和浏览器默认的外观有所不同！

```
<table align="center" summary="我们专门进口来自宇宙其他星球的有机水果 :D">
  <caption>史上最贵水果</caption>

  <thead>
    <tr>
      <th width="200">水果名称</th>
      <th>原产地</th>
      <th width="150">库存量 (斤)</th>
      <th width="150">价格 (每斤)</th>
    </tr>
  </thead>

  <tbody>
    <tr>
      <td>有机磁力火龙果</td>
      <td>水星</td>
      <td>348</td>
      <td>100 万元</td>
    </tr>

    <tr>
      <td>有机离散榴莲</td>
      <td>火星</td>
      <td>850</td>
      <td>200 万元</td>
    </tr>

    <tr>
      <td>有机共轭山竹果</td>
      <td>飞马座星群</td>
      <td>1516</td>
      <td>500 万元</td>
    </tr>
  </tbody>

</table>
```

此外，通过设置 align 属性值为 center，使得表格整体**居中对齐**，并可在 table 元素中设置 summary 属性值为**解释此表格用途**的文字段落，为整个表格程序代码加上特定形式的说明。

5.32 折叠式面板

本节的程序代码都来自于 Ch05-32-accordion.html 范例文件。

图 5-147 是一个由 3 个**按钮**和 3 个**文本块**所构成的**折叠式面板**。

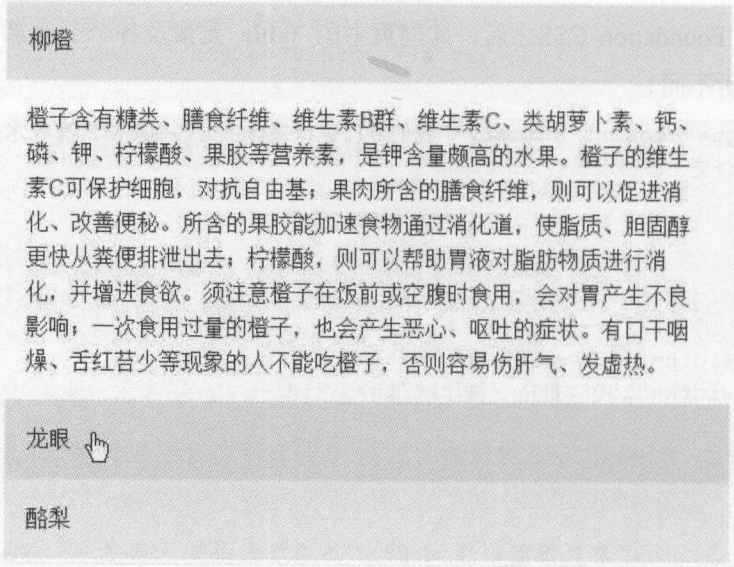


图 5-147　折叠式面板

单击如图 5-147 所示的**第二个**按钮，可看到**第二个**面板被展开了，如图 5-148 所示。

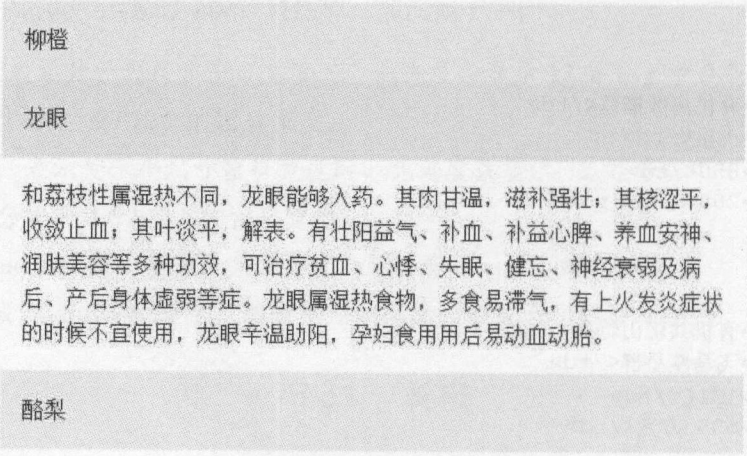


图 5-148　第二个面板被展开了

```
<ul class="accordion" data-accordion>
 <li class="accordion-navigation">
   <a href="#panel1a">柳橙</a>

   <div id="panel1a" class="content active">
     橙子含有糖类、膳食纤维、维生素 B 群、维生素 C、类胡萝卜素、钙、磷、钾、柠
     檬酸、果胶等营养素，是钾含量颇高的水果。橙子的维生素 C 可保护细胞，对抗自
     由基；果肉所含的膳食纤维，则可以促进消化、改善便秘。所含的果胶能加速食物通
     过消化道，使脂质、胆固醇更快从粪便排泄出去；柠檬酸，则可以帮助胃液对脂肪物
     质进行消化，并增进食欲。须注意橙子在饭前或空腹时食用，会对胃产生不良影响；
     一次食用过量的橙子，也会产生恶心、呕吐的症状。有口干咽燥、舌红苔少等现象的
     人不能吃橙子，否则容易伤肝气、发虚热。
```

```
      </div>
    </li>

    <li class="accordion-navigation">
      <a href="#panel2a">龙眼</a>
      <div id="panel2a" class="content">
        和荔枝性属湿热不同，龙眼能够入药。其肉甘温，滋补强壮；其核涩平，收敛止血；
        其叶淡平，解表。有壮阳益气、补血、补益心脾、养血安神、润肤美容等多种功效，
        可治疗贫血、心悸、失眠、健忘、神经衰弱及病后、产后身体虚弱等症。龙眼属湿热
        食物，多食易滞气，有上火发炎症状的时候不宜使用，龙眼辛温助阳，孕妇食用用后
        易动血动胎。
      </div>
    </li>

    <li class="accordion-navigation">
      <a href="#panel3a">酪梨</a>

      <div id="panel3a" class="content">
        一般水果主要提供的营养素是糖类，所以大部分水果吃起来是甜的。但酪梨含糖量很
        低，反而脂肪含量很高，所以若有吃过酪梨的话，你会发现它吃起来没什么甜味，完
        全不像水果。一般水果每 100 公克仅含 0.1~0.2 公克脂肪，酪梨的脂肪却高达 5.9~7.8
        公克，约一般水果的 30~78 倍。所以「脂肪」才是酪梨所能提供的最独特营养素，
        且因酪梨所含的脂肪大部分都是有益健康的单元不饱和脂肪酸和与多元不饱和脂肪，
        所以酪梨是非常好的好脂肪来源食物。<br>
        因为虽然酪梨属于脂肪类食物，但大部分的吃法是把它当水果一样、不经烹调新鲜食
        用，所以除好脂肪外，还可充分保留其中的维生素、矿物质、纤维和植物性营养素。
        且因为它本身富含脂肪，所以还能帮助脂溶性营养素 ADEK 等的吸收。
      </div>
    </li>
  </ul>

  <p style="height: 50px"></p>
```

上述折叠式面板，主要在 ul 元素中设置 class 属性值为 accordion，并设置无属性值的 data-accordion 属性。在 ul 元素里的每个 li 元素中，再设置 class 属性值为 accordion-navigation。在上述的每个 li 元素里面，还有如下的子元素。

- href 属性值为 #panel1a 的 a 元素。
- id 属性值为 panel1a、class 属性值为 content active 的 div 元素。

上述 a 元素的 href 属性值，必须刚好是【#】号加上**后续紧连**的 div 元素的 id 属性值才行。

图 5-149 所示的折叠式面板看起来和**前一个范例**的折叠式面板相似，但是实现的元素并不相同，而且宽度也略有差别。

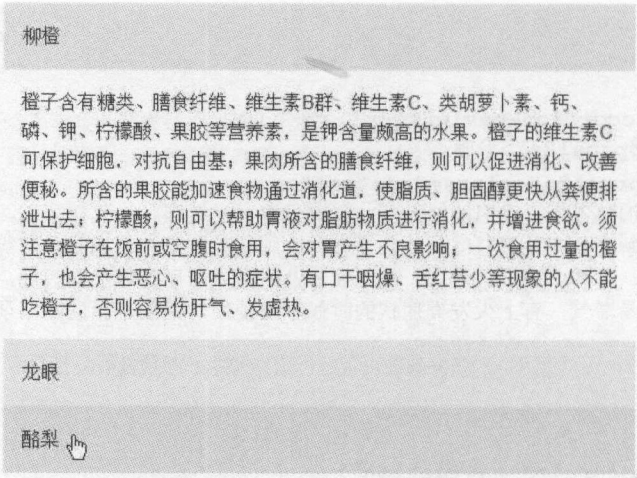


图 5-149　实现元素不一样的折叠式面板

单击图 5-149 中的**第三个**按钮，可以看到**第三个**面板被展开了，如图 5-150 所示。

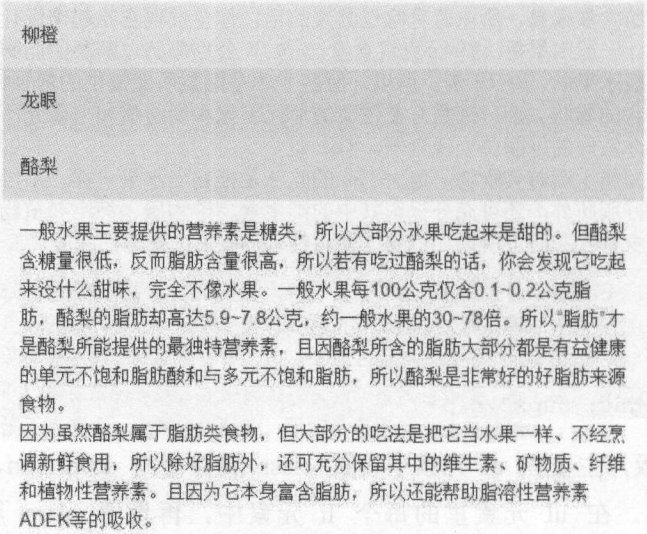


图 5-150　第三个面板被展开了

```
<dl class="accordion" data-accordion>
  <dd class="accordion-navigation">
    <a href="#panel1b">柳橙</a>

    <div id="panel1b" class="content active">
      橙子含有糖类、膳食纤维、维生素 B 群、维生素 C、类胡萝卜素、钙、磷、钾、
      柠檬酸、果胶等营养素，是钾含量颇高的水果。橙子的维生素 C 可保护细胞，对
      抗自由基；果肉所含的膳食纤维，则可以促进消化、改善便秘。所含的果胶能加速
      食物通过消化道，使脂质、胆固醇更快从粪便排泄出去；柠檬酸，则可以帮助胃液
      对脂肪物质进行消化，并增进食欲。须注意橙子在饭前或空腹时食用，会对胃产生
      不良影响；一次食用过量的橙子，也会产生恶心、呕吐的症状。有口干咽燥、舌红
      苔少等现象的人不能吃橙子，否则容易伤肝气、发虚热。
```

```
      </div>
    </dd>

    <dd class="accordion-navigation">
      <a href="#panel2b">龙眼</a>

      <div id="panel2b" class="content">
        和荔枝性属湿热不同，龙眼能够入药。其肉甘温，滋补强壮；其核涩平，收敛止血；
        其叶淡平，解表。有壮阳益气、补血、补益心脾、养血安神、润肤美容等多种功效，
        可治疗贫血、心悸、失眠、健忘、神经衰弱及病后、产后身体虚弱等症。龙眼属湿
        热食物，多食易滞气，有上火发炎症状的时候不宜使用，龙眼辛温助阳，孕妇食用
        用后易动血动胎。
      </div>
    </dd>

    <dd class="accordion-navigation">
      <a href="#panel3b">酪梨</a>

      <div id="panel3b" class="content">
        一般水果主要提供的营养素是糖类，所以大部分水果吃起来是甜的。但酪梨含糖量
        很低，反而脂肪含量很高，所以若有吃过酪梨的话，你会发现它吃起来没什么甜味，
        完全不像水果。一般水果每 100 公克仅含 0.1~0.2 公克脂肪，酪梨的脂肪却高达
        5.9~7.8 公克，约一般水果的 30~78 倍。所以「脂肪」才是酪梨所能提供的最独特
        营养素，且因酪梨所含的脂肪大部分都是有益健康的单元不饱和脂肪酸和与多元不
        饱和脂肪，所以酪梨是非常好的好脂肪来源食物。<br>
        因为虽然酪梨属于脂肪类食物，但大部分的吃法是把它当水果一样、不经烹调新鲜
        食用，所以除好脂肪外，还可充分保留其中的维生素、矿物质、纤维和植物性营养
        素。且因为它本身富含脂肪，所以还能帮助脂溶性营养素 ADEK 等的吸收。
      </div>
    </dd>
  </dl>

  <p style="height: 100px"></p>
```

在本范例的程序代码结构里，主要通过 dl 元素、dd 子元素和最内层的 div 元素来搭建出折叠式面板。

图 5-151 所示的折叠式面板在**浏览器窗口宽度足够**的情况下会分成**左侧** 3 个和**右侧** 3 个的组合。但是，在这 6 个组合里，只有其中一个组合可以被**展开**，其他的组合会被**自动收合**。

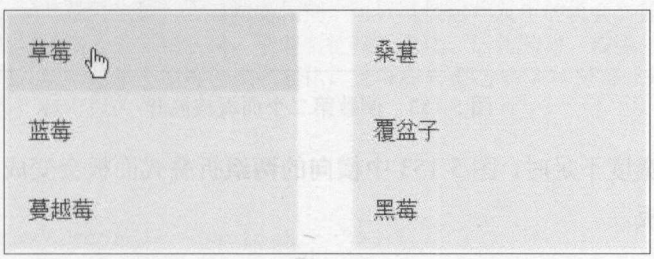


图 5-151　折叠式面板分成左侧 3 个和右侧 3 个的组合

单击图 5-151 中的**第一个**按钮，可以发现**第一个**面板被展开了，如图 5-152 所示。

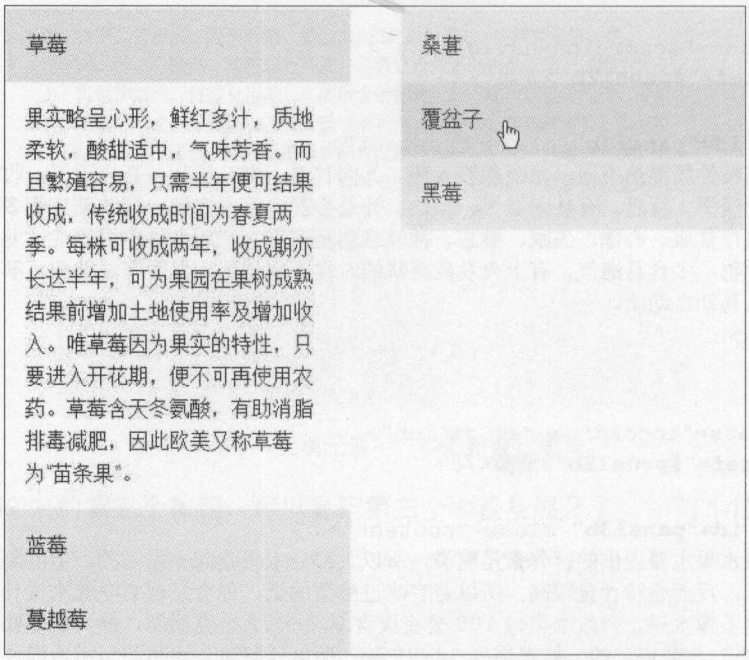


图 5-152　第一个面板被展开

单击图 5-152 中的**倒数第二个**按钮，可以发现**倒数第二个**面板被展开了，如图 5-153 所示。

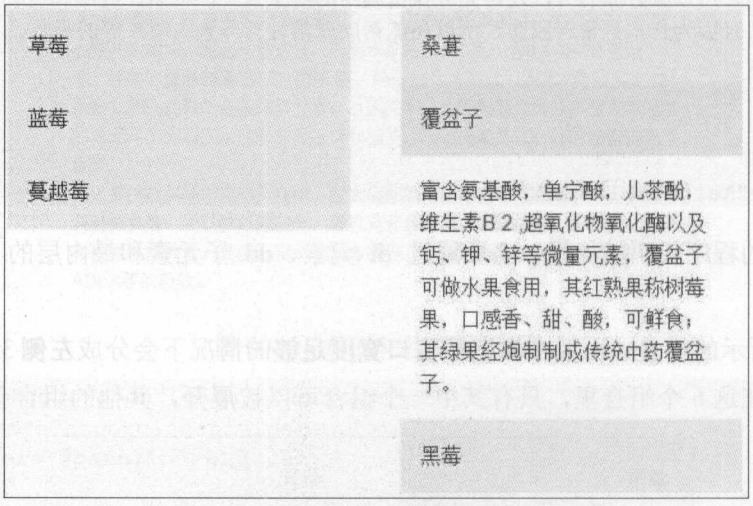


图 5-153　倒数第二个面板被展开

当浏览器窗口宽度不足时，图 5-153 中**横向**的**两组**折叠式面板会变成如图 5-154 所示的**纵向**的**两组**折叠式面板。

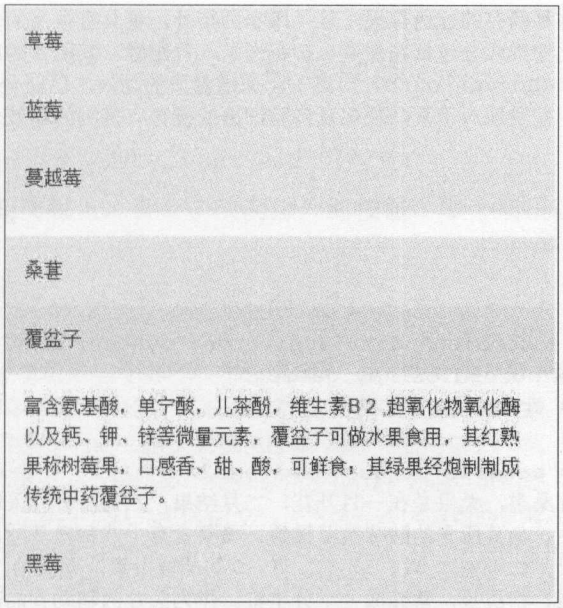


图 5-154　纵向的折叠式面板

```
<ul class="small-block-grid-1 medium-block-grid-3">
  <li>
    <ul class="accordion" data-accordion="myAccordionGroup">
      <li class="accordion-navigation">
        <a href="#panel1c">草莓</a>

        <div id="panel1c" class="content">
          果实略呈心形，鲜红多汁，质地柔软，酸甜适中，气味芳香。而且繁殖容易，
          只需半年便可结果收成，传统收成时间为春夏两季。每株可收成两年，收成期
          亦长达半年，可为果园在果树成熟结果前增加土地使用率及增加收入。唯草莓
          因为果实的特性，只要进入开花期，便不可再使用农药。草莓含天冬氨酸，有
          助消脂排毒减肥，因此欧美又称草莓为“苗条果”。
        </div>
      </li>

      <li class="accordion-navigation">
        <a href="#panel2c">蓝莓</a>

        <div id="panel2c" class="content">
          蓝莓，尤其是野生种，含有抗氧化剂，可以减低癌症发生的机会。蓝莓等浆果
          类植物属于强力抗氧化水果，能够帮忙减缓老化、活化脑力、增强记忆力，国
          外研究发现吃莓果不但可以推迟老化，且每周只要吃 1/2 杯蓝莓或是 1 杯草莓
          的份量，就可以改善记忆力。
        </div>
      </li>

      <li class="accordion-navigation">
        <a href="#panel3c">蔓越莓</a>

        <div id="panel3c" class="content">
```

```
            蔓越莓这种稀有的红色莓果，除了风味独特外，更具有很高的健康价值，主要
            的抗氧化酚类成分包括花青素、初花青素、黄酮醇。其所含的前花青素
            〈proanthocyanidins〉对维护泌尿道健康的功效，已获全球医界证实；同时
            蔓越莓也被发现对于某些胃部及口腔细菌的侵害有抑制的功效。
          </div>
        </li>
      </ul>
    </li>

    <li>
      <ul class="accordion" data-accordion="myAccordionGroup">
        <li class="accordion-navigation">
          <a href="#panel4c">桑葚</a>

          <div id="panel4c" class="content">
            桑葚又称桑果，桑果是在一月开花，二月结果，三月至五月采收，成熟的果实
            为紫黑色。桑葚所含的酸味是苹果酸。桑葚含有十八种氨基酸，同时还含有多
            种维生素，如维生素 B1、B2、C、A、D 和胡萝卜素，葡萄糖，果糖，苹果酸
            以及钙质、铁质等，营养成分十分丰富。作为医疗的辅助食品，有益于控制血
            糖、血压、尿酸、血脂肪。
          </div>
        </li>

        <li class="accordion-navigation">
          <a href="#panel5c">覆盆子</a>

          <div id="panel5c" class="content">
            富含氨基酸，单宁酸，儿茶酚，维生素 B 2 ，超氧化物氧化酶以及钙、钾、锌
            等微量元素，覆盆子可做水果食用，其红熟果称树莓果，口感香、甜、酸，可
            鲜食；其绿果经炮制制成传统中药覆盆子。
          </div>
        </li>

        <li class="accordion-navigation">
          <a href="#panel6c">黑莓</a>

          <div id="panel6c" class="content">
            黑莓的营养功效包括抗衰老、抗氧化、降血压、降血脂、增强心血管功能等，
            这些都跟黑莓所含的维生素 C、多种氨基酸和多酚类养分有关。黑莓还能消暑
            止渴、除痰解酒，而且还有美容效果，外国人宣称黑莓有延年益寿和防癌的功
            效。
          </div>
        </li>
      </ul>
    </li>
  </ul>
```

上述折叠式面板主要是通过设置最上层的 ul 元素的 class 属性值为 small-block-grid-1 medium-block-grid-3，再通过它内部的 li 元素包含原本的**折叠式面板**组件。

5.33 标签面板

本节的程序代码都来自于 Ch05-33-tabs.html 范例文件。

在图 5-155 中的**标签面板**里，有 3 个横向排列的**按钮**，每个按钮对应到一个**文本块**。

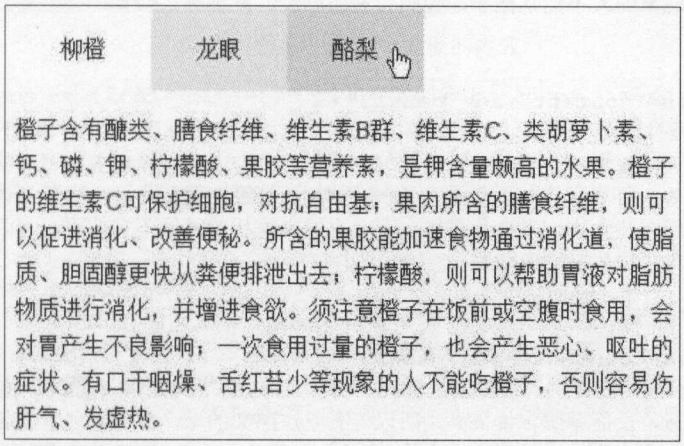


图 5-155 标签面板

单击图 5-155 中的**第三个**按钮，显示出**第三个**面板的内容，如图 5-156 所示。

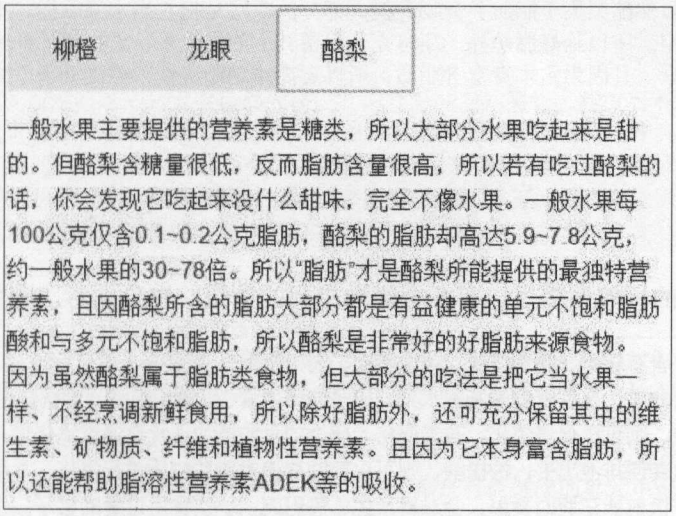


图 5-156 显示出第三个面板的内容

```
<ul class="tabs" data-tab>
   <li class="tab-title active"><a href="#panel1">柳橙</a></li>
   <li class="tab-title"><a href="#panel2">龙眼</a></li>
   <li class="tab-title"><a href="#panel3">酪梨</a></li>
</ul>

<div class="tabs-content">
```

```
    <div class="content active" id="panel1">
      <p>橙子含有糖类、膳食纤维、维生素 B 群、维生素 C、类胡萝卜素、钙、磷、钾、
        柠檬酸、果胶等营养素，是钾含量颇高的水果。橙子的维生素 C 可保护细胞，对
        抗自由基；果肉所含的膳食纤维，则可以促进消化、改善便秘。所含的果胶能加速
        食物通过消化道，使脂质、胆固醇更快从粪便排泄出去；柠檬酸，则可以帮助胃液
        对脂肪物质进行消化，并增进食欲。须注意橙子在饭前或空腹时食用，会对胃产生
        不良影响；一次食用过量的橙子，也会产生恶心、呕吐的症状。有口干咽燥、舌红
        苔少等现象的人不能吃橙子，否则容易伤肝气、发虚热。</p>
    </div>

    <div class="content" id="panel2">
      <p>和荔枝性属湿热不同，龙眼能够入药。其肉甘温，滋补强壮；其核涩平，收敛止
          血；其叶淡平，解表。有壮阳益气、补血、补益心脾、养血安神、润肤美容等多
          种功效，可治疗贫血、心悸、失眠、健忘、神经衰弱及病后、产后身体虚弱等症。
          龙眼属湿热食物，多食易滞气，有上火发炎症状的时候不宜使用，龙眼辛温助阳，
          孕妇食用用后易动血动胎。</p>
    </div>

    <div class="content" id="panel3">
      <p>一般水果主要提供的营养素是糖类，所以大部分水果吃起来是甜的。但酪梨含糖
          量很低，反而脂肪含量很高，所以若有吃过酪梨的话，你会发现它吃起来没什么
          甜味，完全不像水果。一般水果每 100 公克仅含 0.1~0.2 公克脂肪，酪梨的脂肪
          却高达 5.9~7.8 公克，约一般水果的 30~78 倍。所以“脂肪”才是酪梨所能提供
          的最独特营养素，且因酪梨所含的脂肪大部分都是有益健康的单元不饱和脂肪酸
          和与多元不饱和脂肪，所以酪梨是非常好的好脂肪来源食物。<br>
          因为虽然酪梨属于脂肪类食物，但大部分的吃法是把它当水果一样、不经烹调新
          鲜食用，所以除好脂肪外，还可充分保留其中的维生素、矿物质、纤维和植物性
          营养素。且因为它本身富含脂肪，所以还能帮助脂溶性营养素 ADEK 等的吸收。
          </p>
    </div>
</div>

<p></p>
```

在图 5-157 中的标签面板里，有 3 个**纵向**排列的**按钮**，也分别对应到特定的**文本块**。

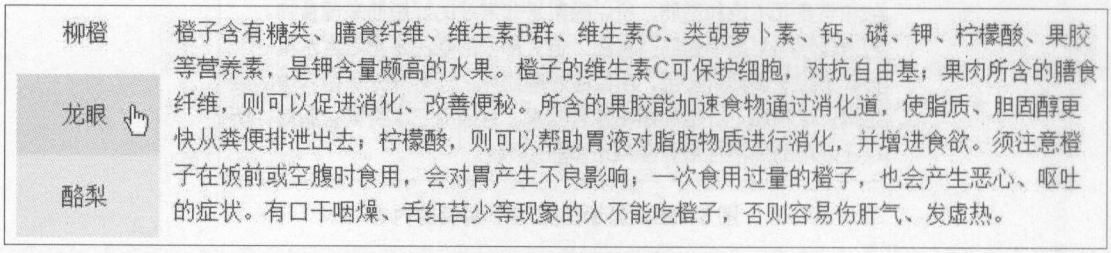


图 5-157　纵向排列按钮的标签面板

单击图 5-157 中的**第二个**按钮，显示出**第二个**面板的内容，如图 5-158 所示。

柳橙

龙眼

酪梨

和荔枝性属湿热不同，龙眼能够入药。其肉甘温，滋补强壮；其核涩平，收敛止血；其叶淡平，解表。有壮阳益气、补血、补益心脾、养血安神、润肤美容等多种功效，可治疗贫血、心悸、失眠、健忘、神经衰弱及病后、产后身体虚弱等症。龙眼属湿热食物，多食易滞气，有上火发炎症状的时候不宜使用，龙眼辛温助阳，孕妇食用用后易动血动胎。

图 5-158　显示出第二个面板的内容

```
<ul class="tabs vertical" data-tab style="width: 100px ; margin-right: 10px">
   <li class="tab-title active"><a href="#panel4">柳橙</a></li>
   <li class="tab-title"><a href="#panel5">龙眼</a></li>
   <li class="tab-title"><a href="#panel6">酪梨</a></li>
</ul>

<div class="tabs-content">
   <div class="content active" id="panel4">
     <p>橙子含有糖类、膳食纤维、维生素 B 群、维生素 C、类胡萝卜素、钙、磷、钾、
        柠檬酸、果胶等营养素，是钾含量颇高的水果。橙子的维生素 C 可保护细胞，
        对抗自由基；果肉所含的膳食纤维，则可以促进消化、改善便秘。所含的果胶能
        加速食物通过消化道，使脂质、胆固醇更快从粪便排泄出去；柠檬酸，则可以帮
        助胃液对脂肪物质进行消化，并增进食欲。须注意橙子在饭前或空腹时食用，会
        对胃产生不良影响；一次食用过量的橙子，也会产生恶心、呕吐的症状。有口干
        咽燥、舌红苔少等现象的人不能吃橙子，否则容易伤肝气、发虚热。</p>
   </div>

   <div class="content" id="panel5">
     <p>和荔枝性属湿热不同，龙眼能够入药。其肉甘温，滋补强壮；其核涩平，收敛止
        血；其叶淡平，解表。有壮阳益气、补血、补益心脾、养血安神、润肤美容等多
        种功效，可治疗贫血、心悸、失眠、健忘、神经衰弱及病后、产后身体虚弱等症。
        龙眼属湿热食物，多食易滞气，有上火发炎症状的时候不宜使用，龙眼辛温助阳，
        孕妇食用用后易动血动胎。</p>
   </div>

   <div class="content" id="panel6">
     <p>一般水果主要提供的营养素是糖类，所以大部分水果吃起来是甜的。但酪梨含糖
        量很低，反而脂肪含量很高，所以若有吃过酪梨的话，你会发现它吃起来没什么
        甜味，完全不像水果。一般水果每 100 公克仅含 0.1~0.2 公克脂肪，酪梨的脂肪
        却高达 5.9~7.8 公克，约一般水果的 30~78 倍。所以“脂肪”才是酪梨所能提供
        的最独特营养素，且因酪梨所含的脂肪大部分都是有益健康的单元不饱和脂肪酸
        和与多元不饱和脂肪，所以酪梨是非常好的好脂肪来源食物。<br>
        因为虽然酪梨属于脂肪类食物，但大部分的吃法是把它当水果一样、不经烹调新
        鲜食用，所以除好脂肪外，还可充分保留其中的维生素、矿物质、纤维和植物性
        营养素。且因为它本身富含脂肪，所以还能帮助脂溶性营养素 ADEK 等的吸收。
        </p>
   </div>
</div>
```

上述程序代码主要通过如下元素的组合构成了一个**标签面板**。

- class 属性值为 tabs（**水平**标签面板）或 tabs vertical（**垂直**标签面板），而且带有无属性值的 data-tab 属性的 ul 元素，并内含 class 属性值为 tab-title 的 li 元素。每个 li 元素又内含了 href 属性值为**【#】号加上特定 div 元素的 id 属性值**。
- class 属性值为 tabs-content 的 div 元素，并内含几个 class 属性值为 content 的 div 元素，而且其 id 属性值为 panel1 至 panel6。其中一个 div 元素的 class 属性值还加注了 active 字样，表示**【处于活动中】**的状态。

5.34 均分版型

本节的程序代码都来自于 Ch05-34-equalizer.html 范例文件。

在图 5-159 中的均分版型里，存在**宽度相同**而且**紧接**在一起的左版面和右版面。

图 5-159 均分版型

当浏览器窗口宽度不足时，图 5-159 中的**横向**版面会变成如图 5-160 所示的**纵向**版面。

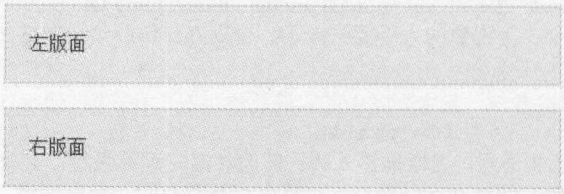


图 5-160 浏览器窗口宽度不足时转变成纵向版面

```
<div class="row" data-equalizer>
   <div class="large-6 columns panel" data-equalizer-watch>
     左版面
   </div>

   <div class="large-6 columns panel" data-equalizer-watch>
     右版面
   </div>
</div>

<p style="height: 50px"></p>
```

上述程序代码是均分版型组件的**第一类型**，主要由如下元素所构成。

- 第一层是 div 元素， class 属性值为 row，并带有无属性值的 data-equalizer 属性。
- 第二层是两个 div 元素，其 class 属性值为 large-6 columns panel，并带有无属性值的 data-equalizer-watch 属性。

在图 5-161 中的均分版面里，存在**左侧**、**中间**和**右侧**的主版面，各个版面之间有一些**间距**。每个**主**版面，其实可如同**左侧**主版面一样再内含几个**纵向**的副版面。

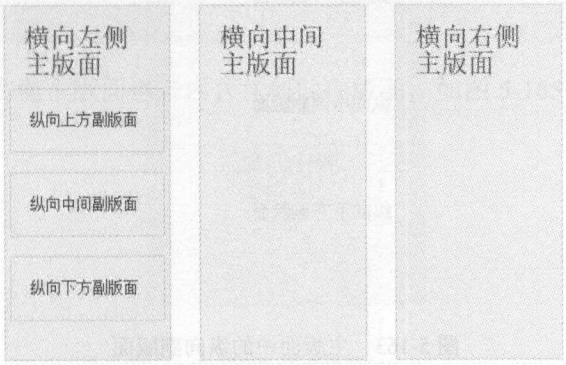


图 5-161　左中右主版面及其上中下副版面

当浏览器窗口宽度不足时，图 5-161 中的**横向版面**会被变成如图 5-162 所示的**纵向版面**。

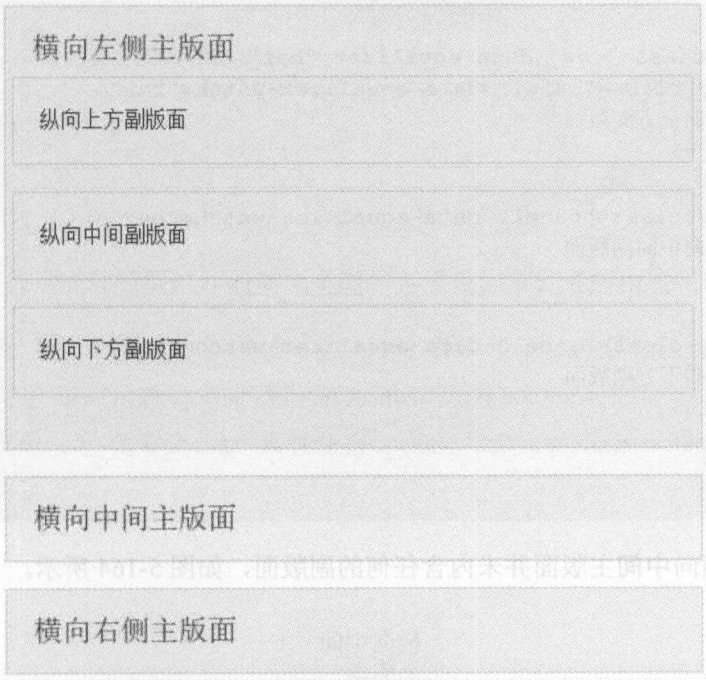


图 5-162　浏览器窗口宽度不足时转变成纵向版面

图 5-163 中的横向**左侧**主版面还内含**纵向**排列的上方、中间和下方共 3 个副版面，副版面之间也都有一些间距。

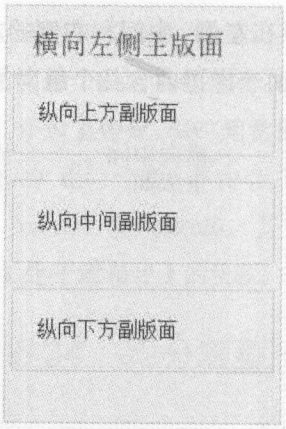


图 5-163　主版面中的纵向副版面

```
<div class="row" data-equalizer="foo">
   <div class="medium-4 columns">
     <div class="panel" data-equalizer-watch="foo">
       <h3>横向左侧主版面</h3>

       <div class="row" data-equalizer="bar">
         <div class="panel" data-equalizer-watch="bar">
           纵向上方副版面
         </div>

         <div class="panel" data-equalizer-watch="bar">
           纵向中间副版面
         </div>

         <div class="panel" data-equalizer-watch="bar">
           纵向下方副版面
         </div>
       </div>
   </div>
</div>
```

在本范例中横向**中间**主版面并未内含任何的副版面，如图 5-164 所示。

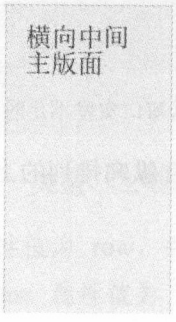


图 5-164　横向中间主版面

```
<div class="medium-4 columns">
   <div class="callout panel" data-equalizer-watch="foo">
     <h3>横向中间主版面</h3>
   </div>
</div>
```

在本范例中横向**右侧**主版面也未内含任何的副版面，如图 5-165 所示。

图 5-165 横向右侧主版面

```
<div class="medium-4 columns">
   <div class="panel" data-equalizer-watch="foo">
     <h3>横向右侧主版面</h3>
   </div>
</div>
</div> <!-- class="row" end -->
```

上述程序代码是均分版型组件的**第二类型**，主要由如下元素所构成。

- 第一层是 div 元素，class 属性值为 row，data-equalizer 属性值为 foo。
- 第二层也是 div 元素，class 属性值为 medium-4 columns。
- 第三层也是 div 元素，class 属性值为 panel，data-equalizer-watch 属性值为 foo。

第 6 章　Yahoo's Pure 的学习与使用

Yahoo's Pure 和前两章所提及的 RWD Framework 比起来，轻巧而易用，待读者亲自操作测试过，便会有同样的感受了。

Yahoo's Pure 在官网上的宣传说明，令人觉得非常谦虚。在笔者编写此书的时间点上，看到其官网上形容 Yahoo's Pure是以“A set of small, responsive CSS modules that you can use in every web project.”寥寥一句的简单描述而已，也就是简单告知用户，可将**一组轻巧的 Yahoo's Pure Framework 自适应 CSS 模块应用于任何的网站 / 网页项目中。**

6.1　Yahoo's Pure 简易应用方式

其使用方式相当简单，只要在特定网页当中加入如下的语句即可：

```
    <link rel="stylesheet" href="http://yui.yahooapis.com/pure/0.6.0/
pure-min.css">
```

读者也可考虑将上述的 http://yui.yahooapis.com/pure/0.6.0/**pure-min.css** 文件下载到自己存放特定网页文件的文件夹中的特定子文件夹里（例如：css），并将上述语句改成如下语句（可参考范例文件 **Ch06-01-index.html**）：

```
    <link rel="stylesheet" href="css/pure/0.6.0/pure-min.css">
```

6.2　版型网格和最小容器单元

本节的程序代码都来自于 Ch06-02-grids-and-units.html 范例文件。

从图 6.1 的 4 组横向版型来看：

- 第一组版型内含 3 个均分窗口总宽度的 div 元素。
- 第二组版型内含 5 个均分窗口总宽度的 div 元素。
- 第三组版型内含 24 个均分窗口总宽度的 div 元素。
- 第四组版型内含 3 个较矮的均分窗口总宽度的 div 元素。

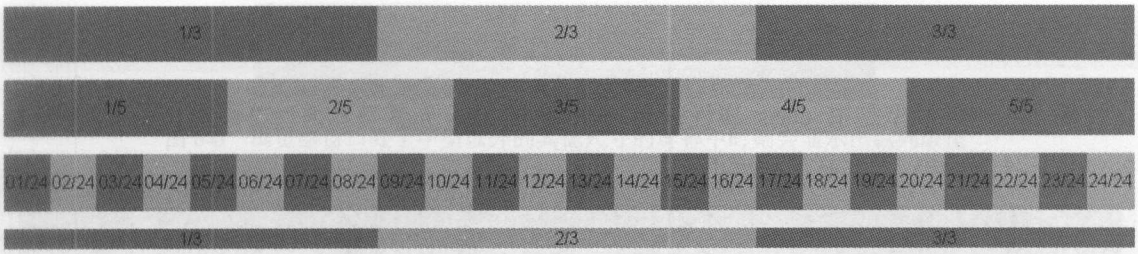


图 6-1 各种版型网格样式

上述版型网格由最上层的 class 属性值为 pure-g 的 div 元素及其内含几个 class 属性值为 pure-u-?-? 的 div 元素所构成。其中，pure-u-?-? 的问号表示的是数字，例如：

- pure-u-1-3 表示上层元素总宽度的**三分之一**。
- pure-u-1-5 表示上层元素总宽度的**五分之一**。
- pure-u-1-24 表示上层元素总宽度的**二十四分之一**。
- **pure-u-1 pure-u-md-1-3** 是指在中型尺寸（md，medium）的窗口宽度时，表示上层元素总宽度的**三分之一**；在其余尺寸的窗口宽度时，表示上层元素总宽度的**全部**。

图 6-2 中的版型是由 class 属性值为 pure-g 的 div 元素及其内含 3 个 class 属性值为 pure-u-1-3 的 div 子元素所构成的。

图 6-2 3 个 class 属性为 pure-u-1-3 的 div 子元素构成的版型网格样式

```
<div class="pure-g">
   <div class="pure-u-1-3"><p>1/3</p></div>
   <div class="pure-u-1-3"><p>2/3</p></div>
   <div class="pure-u-1-3"><p>3/3</p></div>
</div>

<p></p>
```

图 6-3 中的版型是由 class 属性值为 pure-g 的 div 元素及其内含 5 个 class 属性值为 pure-u-1-5 的 div 子元素所构成的。

图 6-3 5 个 class 属性为 pure-u-1-5 的 div 子元素构成的版型网格样式

```
<div class="pure-g">
   <div class="pure-u-1-5"><p>1/5</p></div>
   <div class="pure-u-1-5"><p>2/5</p></div>
   <div class="pure-u-1-5"><p>3/5</p></div>
   <div class="pure-u-1-5"><p>4/5</p></div>
   <div class="pure-u-1-5"><p>5/5</p></div>
</div>
```

```
<p></p>
```

图 6-4 中的版型是由 class 属性值为 pure-g 的 div 元素及其内含 24 个 class 属性值为 pure-u-1-24 的 div 子元素所构成的。

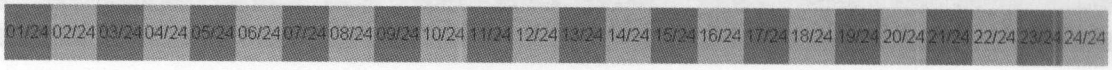


图 6-4　24 个 class 属性为 pure-u-1-24 的 div 子元素构成的版型网格样式

```
<div class="pure-g">
    <div class="pure-u-1-24"><p>01/24</p></div>
    <div class="pure-u-1-24"><p>02/24</p></div>
    <div class="pure-u-1-24"><p>03/24</p></div>
    <div class="pure-u-1-24"><p>04/24</p></div>
    <div class="pure-u-1-24"><p>05/24</p></div>
    <div class="pure-u-1-24"><p>06/24</p></div>
    <div class="pure-u-1-24"><p>07/24</p></div>
    <div class="pure-u-1-24"><p>08/24</p></div>
    <div class="pure-u-1-24"><p>09/24</p></div>
    <div class="pure-u-1-24"><p>10/24</p></div>
    <div class="pure-u-1-24"><p>11/24</p></div>
    <div class="pure-u-1-24"><p>12/24</p></div>
    <div class="pure-u-1-24"><p>13/24</p></div>
    <div class="pure-u-1-24"><p>14/24</p></div>
    <div class="pure-u-1-24"><p>15/24</p></div>
    <div class="pure-u-1-24"><p>16/24</p></div>
    <div class="pure-u-1-24"><p>17/24</p></div>
    <div class="pure-u-1-24"><p>18/24</p></div>
    <div class="pure-u-1-24"><p>19/24</p></div>
    <div class="pure-u-1-24"><p>20/24</p></div>
    <div class="pure-u-1-24"><p>21/24</p></div>
    <div class="pure-u-1-24"><p>22/24</p></div>
    <div class="pure-u-1-24"><p>23/24</p></div>
    <div class="pure-u-1-24"><p>24/24</p></div>
</div>

<p></p>
```

图 6-5 中的版型是由 class 属性值为 pure-g 的 div 元素及其内含 3 个 class 属性值为 pure-u-1 pure-u-md-1-3 的 div 子元素所构成的。

图 6-5　3 个 class 属性为 pure-u-1 pure-u-md-1-3 的 div 子元素构成的版型网格样式

当浏览器窗口处于中型（md，medium）尺寸的屏幕中时，会显示如图 6-5 所示的**横向**版型；当浏览器窗口处于中型**以外**的其他尺寸的屏幕中时，则会显示如图 6-6 所示的**纵向**版型。

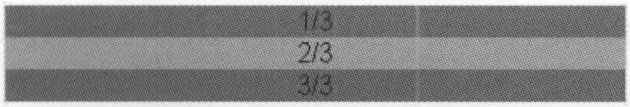


图 6-6 浏览器窗口处于中型以外的其他尺寸的屏幕中时则会显示出纵向版型

```
<div class="pure-g">
   <div class="pure-u-1 pure-u-md-1-3">1/3</div>
   <div class="pure-u-1 pure-u-md-1-3">2/3</div>
   <div class="pure-u-1 pure-u-md-1-3">3/3</div>
</div>
```

6.3 窗体组件

本节的程序代码都来自于 Ch06-03-forms.html 范例文件。

在 form 元素中，设置其 class 属性值为如下几种字样。

- pure-form，表示会被应用 pure CSS 默认的窗体样式。
- pure-form-stacked，表示会被应用 pure CSS 的堆栈型窗体样式。
- pure-form-aligned，表示会被应用 pure CSS 的对齐式窗体样式。

在 form 元素中，会有如下几个应用 pure CSS 类的元素。

- class 属性值为 pure-control-group 或 pure-controls 的 div 元素。
- class 属性值为 pure-input-1-2 的 select 元素。
- class 属性值为 pure-button pure-input-1-2 pure-button-primary、type 属性值为 button 的 button 元素。
- class 属性值为 pure-u-23-24、pure-input-1、pure-input-1-2、pure-input-2-3、pure-input-1-3、pure-input-1-4 或 pure-input-rounded，而且 type 属性值为 text 的 input 元素。
- class 属性值为 pure-g、pure-u-1 pure-u-md-1-3、pure-u-1-4、pure-u-3-4、pure-u-1-2、pure-u-1-8、pure-u-1-5 或 pure-u-2-5 的 div 元素。
- class 属性值为 pure-checkbox 或 pure-radio 的 label 元素。
- class 属性值为 pure-button 或 pure-button pure-button-primary，而且 type 属性值为 submit 的 button 元素。

图 6-7 中的部分窗体组件里带有两个文本栏、一个多选按钮（复选框）和一个按钮。当插入点进入文本栏内部时，此文本栏会显示出**显眼**的**外框**；当鼠标指针移至按钮的范围内时，则会变成**手掌**形状。

当文本栏内尚未输入任何文字时，会在其范围内以**浅灰色**字样提示用户应该输入哪种性质的资料。

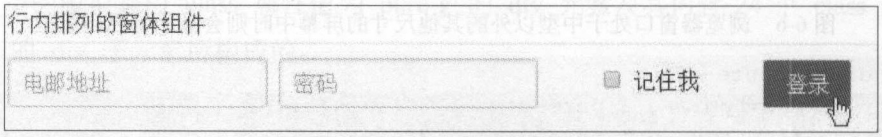


图 6-7　窗体组件

当窗口宽度不足时，窗体组件会自动变成如图 6-8 所示的外观显示。

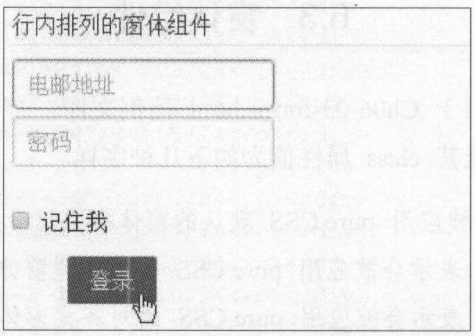


图 6-8　当浏览器窗口宽度不足时窗体组件自动变成纵向显示

```
    <form class="pure-form">
       <fieldset>
         <legend>行内排列的窗体组件</legend>

         <input type="email" placeholder="电邮地址">
         <input type="password" placeholder="密码">

           
         <label for="remember">
           <input id="remember" type="checkbox">记住我
         </label>
           

         <button type="submit" class="pure-button pure-button-primary">登录
</button>
       </fieldset>
    </form>

    <p style="height: 20px"></p>
```

在图 6-9 中的部分窗体组件里，可以看到各个窗体组件（包括标签、文本栏、下拉式菜单、多选按钮和按钮）都是**纵向**排列的。

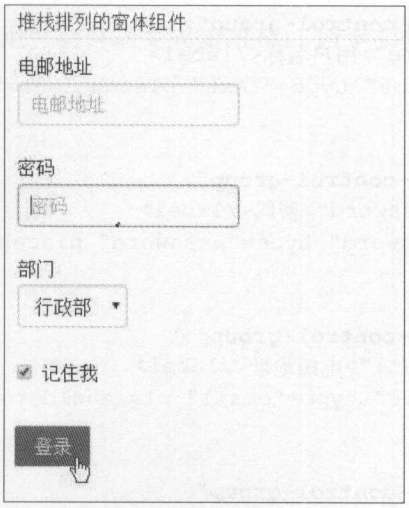


图 6-9 全部纵向排列的窗体组件

```
    <form class="pure-form pure-form-stacked">
        <fieldset>
            <legend>堆栈排列的窗体组件</legend>

            <label for="email">电邮地址</label>
            <input id="email" type="email" placeholder="电邮地址">
            <br>

            <label for="password">密码</label>
            <input id="password" type="password" placeholder="密码">
            <br>

            <label for="department">部门</label>
            <select id="department">
                <option>信息部</option>
                <option>行政部</option>
                <option>人事部</option>
            </select>
            <br>

            <label for="remember" class="pure-checkbox">
                <input id="remember" type="checkbox">记住我
            </label>
            <br>

            <button type="submit" class="pure-button pure-button-primary">登录
</button>
        </fieldset>
    </form>

    <p style="height: 20px"></p>

    <form class="pure-form pure-form-aligned">
        <fieldset>
```

```
        <div class="pure-control-group">
          <label for="name">用户名称</label>
            <input id="name" type="text" placeholder="用户名称">
        </div>

        <div class="pure-control-group">
          <label for="password">密码</label>
          <input id="password" type="password" placeholder="密码">
        </div>

        <div class="pure-control-group">
          <label for="email">电邮地址</label>
          <input id="email" type="email" placeholder="电邮地址">
        </div>

        <div class="pure-control-group">
          <label for="foo">参加的活动</label>
          <input id="foo" type="text" placeholder="参加的活动">
        </div>

        <div class="pure-controls">
          <label for="cb" class="pure-checkbox">
            <input id="cb" type="checkbox">我已经阅读本公司条款
          </label>
          <br>
          <button type="submit" class="pure-button pure-button-primary">提交
</button>
        </div>
      </fieldset>
    </form>

    <p style="height: 20px"></p>
```

在本节的范例程序代码当中，用来表示传统的下拉式菜单的是 select 元素和其内部多个 option 元素所构成的程序代码结构。

在图 6-10 中的部分窗体组件里，由单个**字段集**（field set）元素包含多个**标签和文本栏**与一个**标签和下拉式菜单**的组合。

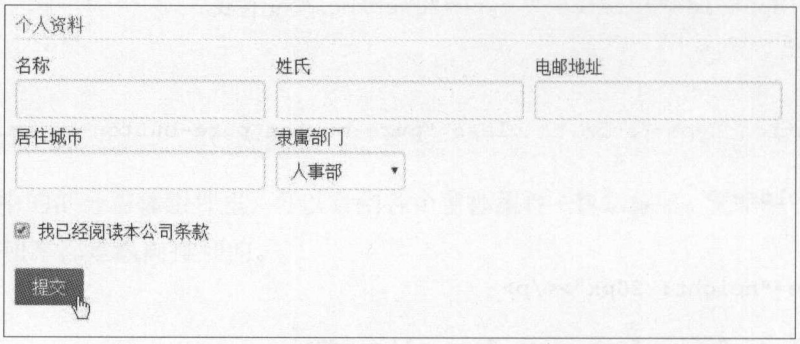


图 6-10　单个字段集元素包含多个标签和文本栏与一个标签和下拉式菜单的组合

当窗口宽度不足时，窗体组件会自动变成如图 6-11 所示的外观显示。

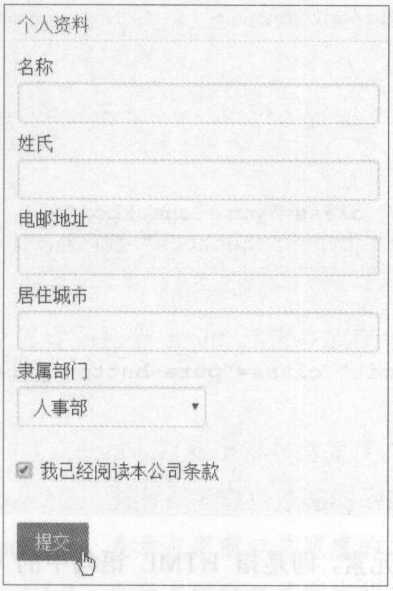


图 6-11　当窗口宽度不足时窗体组件自动变成纵向显示

```
<form class="pure-form pure-form-stacked">
  <fieldset>
    <legend>个人资料</legend>

    <div class="pure-g">
      <div class="pure-u-1 pure-u-md-1-3">
        <label for="first-name">名称</label>
        <input id="first-name" class="pure-u-23-24" type="text">
      </div>

      <div class="pure-u-1 pure-u-md-1-3">
        <label for="last-name">姓氏</label>
        <input id="last-name" class="pure-u-23-24" type="text">
      </div>

      <div class="pure-u-1 pure-u-md-1-3">
        <label for="email">电邮地址</label>
        <input id="email" class="pure-u-23-24" type="email" required>
      </div>

      <div class="pure-u-1 pure-u-md-1-3">
        <label for="city">居住城市</label>
        <input id="city" class="pure-u-23-24" type="text">
      </div>

      <div class="pure-u-1 pure-u-md-1-3">
        <label for="department">隶属部门</label>
        <select id="department" class="pure-input-1-2">
```

```
            <option>信息部</option>
            <option>行政部</option>
            <option>人事部</option>
          </select>
        </div>
      </div>
      <br>

      <label for="terms" class="pure-checkbox">
        <input id="terms" type="checkbox">我已经阅读本公司条款
      </label>
      <br>

      <button type="submit" class="pure-button pure-button-primary">提交
</button>
    </fieldset>
  </form>
  <p style="height: 20px"></p>
```

上述的字段集（field set）元素，即是指 HTML 语句中的 <fieldset> … </fieldset> 元素语句所构成的部分。

在图 6-12 中的部分窗体组件里，可以看到除了没有**字段集**的**标题文字**之外，也没有各文本栏前的**标签文字**，而是直接通过各文本栏内的浅灰色的提示字样来告知用户应该分别输入什么性质的资料，这反而是节省网页版面空间的一种方法。

在本范例中，还有一个以 textarea 元素打造的**多行**文本栏来输入**工作性质与内容**，便于用户输入较多的资料。另外，用户也可动态拖曳多行文本栏的边框，以缩放其输入文字的范围。

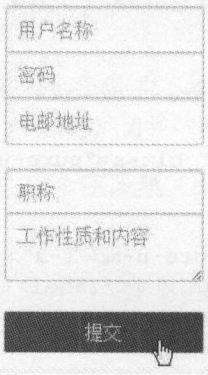


图 6-12　多行文本栏组成的窗体组件

```
  <form class="pure-form">
    <fieldset class="pure-group">
      <input type="text" class="pure-input-1-2" placeholder="用户名称">
      <input type="text" class="pure-input-1-2" placeholder="密码">
      <input type="email" class="pure-input-1-2" placeholder="电邮地址">
    </fieldset>
```

```
      <fieldset class="pure-group">
        <input type="text" class="pure-input-1-2" placeholder="职称">
        <textarea class="pure-input-1-2" placeholder="工作性质和内容"></textarea>
      </fieldset>

      <button type="submit" class="pure-button pure-input-1-2
pure-button-primary">
          提交</button>
    </form>

    <p style="height: 20px"></p>
```

在上述程序代码结构中，可以看到几个 input 元素被放在单个 fieldset 元素中变成一组。在图 6-13 中的多种宽度的文本栏里：

- class 属性值为 pure-input-1，表示占用**整个**窗口总宽度。
- class 属性值为 pure-input-2-3，表示占用窗口总宽度的**三分之二**。
- class 属性值为 pure-input-1-2，表示占用窗口总宽度的**二分之一**。
- class 属性值为 pure-input-1-3，表示占用窗口总宽度的**三分之一**。
- class 属性值为 pure-input-1-4，表示占用窗口总宽度的**四分之一**。

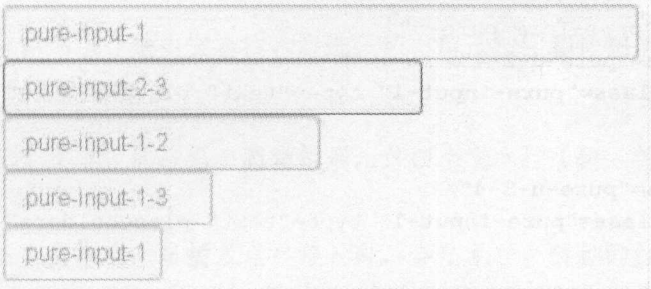


图 6-13　多种宽度的文本栏 1

```
    <form class="pure-form">
      <input class="pure-input-1" type="text" placeholder=".pure-input-1"><br>
      <input class="pure-input-2-3" type="text" placeholder=".pure-input-2-3">
<br>
      <input class="pure-input-1-2" type="text" placeholder=".pure-input-1-2">
<br>
      <input class="pure-input-1-3" type="text" placeholder=".pure-input-1-3">
<br>
      <input class="pure-input-1-4" type="text" placeholder=".pure-input-1-4">
<br>
    </form>

    <p style="height: 50px"></p>
```

在图 6-14 中的多种宽度的文本栏里：

- class 属性值为 pure-input-1-4，表示占用窗口总宽度的**四分之一**。
- class 属性值为 pure-input-3-4，表示占用窗口总宽度的**四分之三**。
- class 属性值为 pure-input-1-2，表示占用窗口总宽度的**二分之一**。
- class 属性值为 pure-input-1-8，表示占用窗口总宽度的**八分之一**。
- class 属性值为 pure-input-1-4，表示占用窗口总宽度的**四分之一**。
- class 属性值为 pure-input-1-5，表示占用窗口总宽度的**五分之一**。
- class 属性值为 pure-input-2-5，表示占用窗口总宽度的**五分之二**。
- class 属性值为 pure-input-1，表示占用**整个**窗口总宽度。

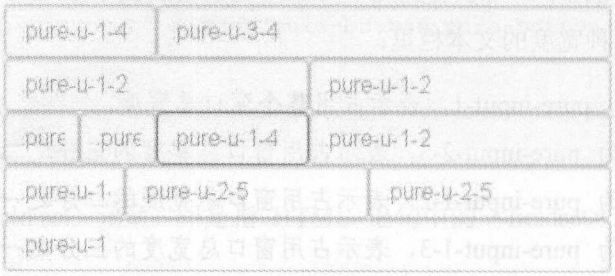


图 6-14　多种宽度的文本栏 2

```
<form class="pure-form pure-g">
    <div class="pure-u-1-4">
      <input class="pure-input-1" type="text" placeholder=".pure-u-1-4">
    </div>

    <div class="pure-u-3-4">
      <input class="pure-input-1" type="text" placeholder=".pure-u-3-4">
    </div>

    <div class="pure-u-1-2">
      <input class="pure-input-1" type="text" placeholder=".pure-u-1-2">
    </div>

    <div class="pure-u-1-2">
      <input class="pure-input-1" type="text" placeholder=".pure-u-1-2">
    </div>

    <div class="pure-u-1-8">
      <input class="pure-input-1" type="text" placeholder=".pure-u-1-8">
    </div>

    <div class="pure-u-1-8">
      <input class="pure-input-1" type="text" placeholder=".pure-u-1-8">
    </div>

    <div class="pure-u-1-4">
      <input class="pure-input-1" type="text" placeholder=".pure-u-1-4">
```

```
    </div>

    <div class="pure-u-1-2">
      <input class="pure-input-1" type="text" placeholder=".pure-u-1-2">
    </div>

    <div class="pure-u-1-5">
      <input class="pure-input-1" type="text" placeholder=".pure-u-1-5">
    </div>

    <div class="pure-u-2-5">
      <input class="pure-input-1" type="text" placeholder=".pure-u-2-5">
    </div>

    <div class="pure-u-2-5">
      <input class="pure-input-1" type="text" placeholder=".pure-u-2-5">
    </div>

    <div class="pure-u-1">
      <input class="pure-input-1" type="text" placeholder=".pure-u-1">
    </div>
  </form>

  <p style="height: 50px"></p>
```

图 6-15 中的部分窗体组件是由文本栏、多选按钮、单选按钮和按钮所组成的。其中，在本范例中的文本栏有如下的特性。

- 第一个至第三个文本栏的边框为**圆角矩形**，第四个文本栏（输入**关键字**）的边框为**椭圆形**。
- 第一个或第四个文本栏，当**插入**点被移入时，会显示一个醒目的红色边框，以提示用户此为当前的**焦点**所在地。
- 第二个文本栏是处于【停用中】状态，第三个文本栏是【只读】型的文本栏，所以不允许用户在其内输入资料。

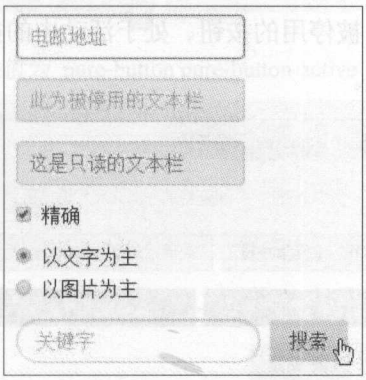


图 6-15　由文本栏、多选按钮、单选按钮和按钮所组成的窗体组件

```
    <form class="pure-form">
        <input type="email" placeholder="电邮地址" required>
        <p></p>

        <input type="text" placeholder="此为被停用的文本栏" disabled>
        <p></p>

        <input type="text" value="这是只读的文本栏" readonly>
        <p></p>

        <label for="option-one" class="pure-checkbox">
          <input id="option-one" type="checkbox" value="">精确
        </label>
        <p></p>

        <label for="option-two" class="pure-radio">
    <input id="option-two" type="radio" name="optionsRadios"
     value="option1" checked>以文字为主
        </label>

        <label for="option-three" class="pure-radio">
          <input id="option-three" type="radio" name="optionsRadios"
value="option2">
              以图片为主
        </label>
        <p></p>

        <input type="text" class="pure-input-rounded" placeholder="关键字">

        <button type="submit" class="pure-button">搜索</button>
    </form>
```

6.4 按钮

本节的程序代码都来自于 Ch06-04-buttons.html 范例文件。

在图 6-16 中的各行里，都有左侧 a 元素所构的按钮和右侧 button 元素所构成的按钮。从上而下的各行分别为一般按钮、被停用的按钮、处于活动中的按钮、被应用 primary 颜色样式的按钮、内含内置小图标的按钮。

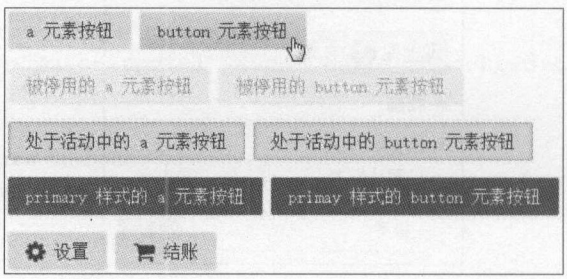


图 6-16 各种样式的按钮

在 a 或 button 元素中，可以设置 class 属性值为 pure-button，并可在 class 属性值中额外加注如下字样。

- pure-button-disabled，使得此按钮显示【停用中】状态的外观。
- pure-button-active，使得此按钮显示【处于活动中】状态的外观。
- pure-button-primary，使得此按钮显示 primary 颜色样式的外观。

图 6-17 为 class 属性值为 pure-button 的一般按钮。

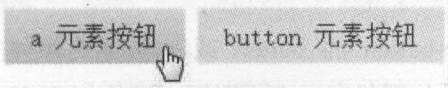


图 6-17 class 属性值为 pure-button 的一般按钮

```
    <a class="pure-button" href="#">a 元素按钮</a>
    <button class="pure-button">button 元素按钮</button>

    <p></p>
```

图 6-18 为 class 属性值为 pure-button pure-button-disabled 的被停用的按钮。鼠标指针移至被停用的按钮范围内，就会变成禁用符号。

图 6-18 class 属性值为 pure-button pure-button-disabled 的被停用的按钮

```
    <a class="pure-button pure-button-disabled" href="#">被停用的 a 元素按钮</a>
    <button class="pure-button pure-button-disabled">被停用的 button 元素按钮
</button>

    <p></p>
```

图 6-19 为 class 属性值为 pure-button pure-button-active 的**处于活动中**的按钮。

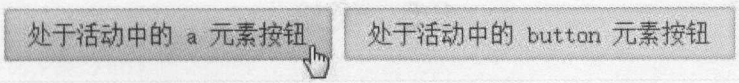


图 6-19 class 属性值为 pure-button pure-button-active 的处于活动中的按钮

```
    <a class="pure-button pure-button-active" href="#">处于活动中的 a 元素按钮</a>
    <button class="pure-button pure-button-active">处于活动中的 button 元素按钮
</button>

    <p></p>
```

图 6-20 为 class 属性值为 pure-button pure-button-primary 并应用了 primary 颜色样式的按钮。

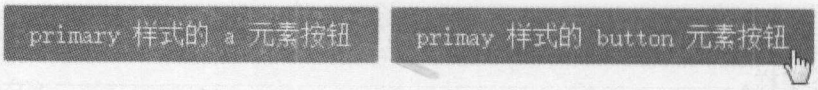


图 6-20　class 属性值为 pure-button pure-button-primary 并应用了 primary 颜色样式的按钮

```
    <a class="pure-button pure-button-primary" href="#">primary 样式的 a 元素按钮
</a>
    <button class="pure-button pure-button-primary">primary 样式的 button 元素按钮
</button>

    <p></p>
```

在 class 属性值为 pure-button 的 button 或 a 元素内额外安排一个 class 属性值为 fa 加上 fa-lg 或 fa-sm，再安排如下代表一个小图标字样的 i 元素来生成带有图案的按钮：

- fa-cog，代表设置含义的图案的 pure CSS 类名称。
- fa-shopping-cart，代表**购物车**含义的图案的 pure CSS 类名称。

其中，fa-lg 可使得上述图案变**大**，而 fa-sm 可使得上述图案变**小**。图 6-21 所示为带有图案的按钮。

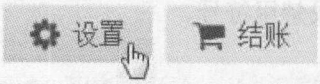


图 6-21　具有图案的按钮

```
    <button class="pure-button">
      <i class="fa fa-cog fa-lg"></i> 设置
    </button>

    <a class="pure-button" href="#">
      <i class="fa fa-shopping-cart fa-lg"></i> 结账
    </a>
```

6.5　表格

本节的程序代码都来自于 Ch06-05-tables.html 范例文件。

首先，在 style 元素之内设置如下自定义的 CSS 规则语句，使得特定或所有表格元素的**整体结构**居中对齐（margin: auto），而其**内部文字**居中对齐 (text-align: center)：

```
table { margin: auto ; text-align: center ; }
```

在网页中显示的表格默认是由 table 元素搭建而成的。若在 table 元素当中设置 class 属性值为 pure-table，则可使得此 table 元素具有 pure CSS 所支持的表格外观。table 元素的 class 属性值可再额外加注如下字样。

- pure-table-bordered，可使得其表格元素具有 pure CSS 的单元格网格线样式。

- pure-table-horizontal，可使得其表格元素具有 pure CSS 的行网格线样式。

可在表格内的 tr 元素中设置 class 属性值为 pure-table-odd，使表格具有**间隔行底纹**和**列网格线**。

在图 6-22 中的表格组件里，没有**行**网格线，只有**列**网格线，内含表格标题，是具有 4 行 4 列的表格结构。其中，第一行具有**浅灰背景**，是包含了 4 个**粗体**的**字段标题**单元格的标题栏。

有列网格线的表格

编号	种类	果肉主色	库存（斤）
1	火龙果	红色	50
2	小玉西瓜	黄色	80
3	樱桃	紫色	120

图 6-22 只有列网格线的表格

```
<style>
   table { margin: auto ; text-align: center ; }
</style>
…
   <table class="pure-table">
<caption>有列网格线的表格</caption>

   <thead>
     <tr>
       <th>编号</th>
       <th>种类</th>
       <th>果肉主色</th>
       <th>库存 (斤)</th>
     </tr>
   </thead>

   <tbody>
     <tr>
       <td>1</td>
       <td>火龙果</td>
       <td>红色</td>
       <td>50</td>
     </tr>

     <tr>
       <td>2</td>
       <td>小玉西瓜</td>
       <td>黄色</td>
       <td>80</td>
     </tr>

     <tr>
       <td>3</td>
```

```
        <td>樱桃</td>
        <td>紫色</td>
        <td>120</td>
      </tr>
    </tbody>
</table>

<p style="height: 10px"></p>
```

在图 6-23 中的表格组件里，**每个**单元格都具有**完整**的**网格线**，也内含表格标题，是具有 4 行 4 列的表格结构。其中，第一行是包含 4 个**粗体**的**字段标题**单元格的标题栏。

有单元格网格线的表格

编号	种类	果肉主色	库存（斤）
1	火龙果	红色	50
2	小玉西瓜	黄色	80
3	樱桃	紫色	120

图 6-23　具有完整网格线的表格

```
<table class="pure-table pure-table-bordered">
    <caption>有单元格网格线的表格</caption>

    <thead>
      <tr>
        <th>编号</th>
        <th>种类</th>
        <th>果肉主色</th>
        <th>库存 (斤)</th>
      </tr>
    </thead>

    <tbody>
      <tr>
        <td>1</td>
        <td>火龙果</td>
        <td>红色</td>
        <td>50</td>
      </tr>

      <tr>
        <td>2</td>
        <td>小玉西瓜</td>
        <td>黄色</td>
        <td>80</td>
      </tr>

      <tr>
        <td>3</td>
        <td>樱桃</td>
```

```
        <td>紫色</td>
        <td>120</td>
      </tr>
    </tbody>
  </table>

  <p style="height: 10px"></p>
```

在图 6-24 中的表格组件里，只有**行**网格线，并没有**列网格线**，内含表格标题，是具有 4 行 4 列的表格结构。其中，第一行是包含 4 个**粗体**的**字段标题**单元格的标题栏。

有列网格线的表格

编号	种类	果肉主色	库存（斤）
1	火龙果	红色	50
2	小玉西瓜	黄色	80
3	樱桃	紫色	120

图 6-24　只有行网格线的表格

```
  <table class="pure-table pure-table-horizontal">
    <caption>有行网格线的表格</caption>

    <thead>
      <tr>
        <th>编号</th>
        <th>种类</th>
        <th>果肉主色</th>
        <th>库存 (斤)</th>
      </tr>
    </thead>

    <tbody>
      <tr>
        <td>1</td>
        <td>火龙果</td>
        <td>红色</td>
        <td>50</td>
      </tr>

      <tr>
        <td>2</td>
        <td>小玉西瓜</td>
        <td>黄色</td>
        <td>80</td>
      </tr>

      <tr>
        <td>3</td>
        <td>樱桃</td>
```

```
        <td>紫色</td>
        <td>120</td>
      </tr>
    </tbody>
</table>

<p style="height: 10px"></p>
```

在图 6-25 中的表格组件里，只有列网格线，并无行网格线，但是除了第一行有**较深**灰背景之外，其余各行相间着**较浅**灰背景。此表格组件内含表格标题，是具有 5 行 4 列的表格结构。其中，第一行是包含 4 个**粗体**的**字段标题**单元格的标题栏。

有间隔行底纹 & 列网格线的表格

编号	种类	果肉主色	库存（斤）
1	火龙果	红色	50
2	小玉西瓜	黄色	80
3	樱桃	紫色	120
4	酪梨	绿色	60

图 6-25　有间隔行底纹和列网格线的表格

```
<table class="pure-table">
    <caption>有间隔行底纹 & 列网格线的表格</caption>

    <thead>
      <tr>
        <th>编号</th>
        <th>种类</th>
        <th>果肉主色</th>
        <th>库存 (斤)</th>
      </tr>
    </thead>

    <tbody>
      <tr class="pure-table-odd">
        <td>1</td>
        <td>火龙果</td>
        <td>红色</td>
        <td>50</td>
      </tr>

      <tr>
        <td>2</td>
        <td>小玉西瓜</td>
        <td>黄色</td>
        <td>80</td>
      </tr>
```

```
        <tr class="pure-table-odd">
          <td>3</td>
          <td>樱桃</td>
          <td>紫色</td>
          <td>120</td>
        </tr>

        <tr>
          <td>4</td>
          <td>酪梨</td>
          <td>绿色</td>
          <td>60</td>
        </tr>
      </tbody>
    </table>
```

6.6 菜单

本节的程序代码都来自于 Ch06-06-menus.html 范例文件。

在图 6-26 中的菜单里，具有**粗体**的标题栏文字（例如：主要城市、其他介绍）和可选择的各选项。

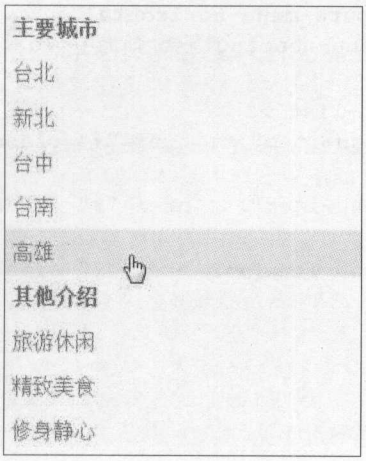


图 6-26 具有粗体标题栏和选项的菜单

```
    <div class="pure-menu custom-restricted-width">
      <!--<span class="pure-menu-heading">主要城市</span> -->
      <ul class="pure-menu-list">
        <li class="pure-menu-heading"><b>主要城市</b></li>
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">台北
</a></li>
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">新北
</a></li>
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">台中
</a></li>
```

```
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">台南
</a></li>
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">高雄
</a></li>
        <li class="pure-menu-heading"><b>其他介绍</b></li>
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">旅游休闲
</a></li>
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">精致美食
</a></li>
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">修身静心
</a></li>
      </ul>
    </div>

    <p style="height: 50px"></p>
```

在图 6-27 中的**横向**菜单里，带有不可选择的**粗体**的标题文字（例如：航海王）和可选择的项目。

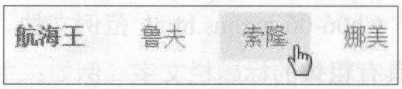


图 6-27　具有不可选择和可选择选项的横向菜单

```
    <div class="pure-menu pure-menu-horizontal">
      <span class="pure-menu-heading"><b>航海王</b></span>

      <ul class="pure-menu-list">
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">鲁夫
</a></li>
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">索隆
</a></li>
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">娜美
</a></li>
      </ul>
    </div>

    <p style="height: 50px"></p>
```

在图 6-28 中的菜单里，笔者故意凸显出各选项可以是**被选取**的、**一般**的或**被停用**的选项。

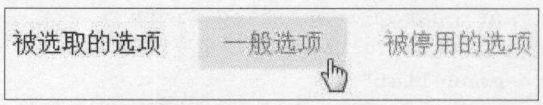


图 6-28　被选取的、一般的和被停用的选项

```
    <div class="pure-menu pure-menu-horizontal">
      <ul class="pure-menu-list">
        <li class="pure-menu-item pure-menu-selected">
          <a href="#" class="pure-menu-link">被选取的项目</a></li>
```

```
        <li class="pure-menu-item"><a href="#" class="pure-menu-link">一般项目
</a></li>

        <li class="pure-menu-item pure-menu-disabled">被停用的项目</li>
      </ul>
    </div>

    <p style="height: 50px"></p>
```

在图 6-29 中的菜单里，【首页】是此菜单的**第一**个选项，而【联络方式】是**第二**个选项。当鼠标指针被移至【联络方式】的范围内时，便会自动打开**第二层**的子菜单（由**电邮地址**、**市内电话**、**留言系统**所构成的子菜单）。

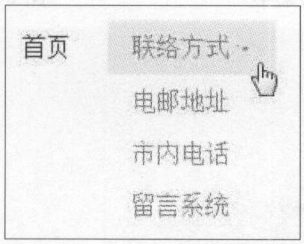


图 6-29　具有主菜单和子菜单的菜单栏

```
    <div class="pure-menu pure-menu-horizontal">
      <ul class="pure-menu-list">
        <li class="pure-menu-item pure-menu-selected"><a href="#"
          class="pure-menu-link">首页</a></li>

        <li class="pure-menu-item pure-menu-has-children pure-menu-allow-hover">
          <a href="#" id="menuLink1" class="pure-menu-link">联络方式</a>

          <ul class="pure-menu-children">
            <li class="pure-menu-item"><a href="#"
              class="pure-menu-link">电邮地址</a></li>
            <li class="pure-menu-item"><a href="#"
              class="pure-menu-link">市内电话</a></li>
            <li class="pure-menu-item"><a href="#"
              class="pure-menu-link">留言系统</a></li>
          </ul>
        </li>
      </ul>
    </div>

    <p style="height: 100px"></p>
```

图 6-30 中的菜单以较局限的范围来显示其**标题栏**和各**选项**，并搭配**滚动条**来显示后续的各选项。

此菜单容纳相当多的选项，往下拖曳滚动条到底部，最下方的选项也都显示出来了，如图6-31所示。

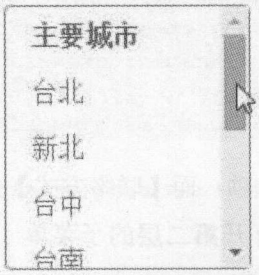


图6-30　带滚动条和选项的菜单

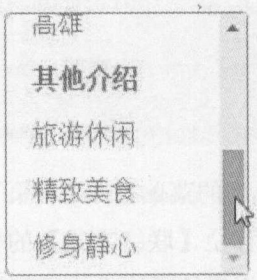


图6-31　拖曳滚动条到菜单底部

```
    <style>
        .custom-restricted
        {
        height: 160px;
        width: 150px;
        border: 1px solid gray;
        border-radius: 4px;
        }
    </style>
    …
    <div class="pure-menu pure-menu-scrollable custom-restricted">
       <ul class="pure-menu-list">
         <li class="pure-menu-heading"><b>主要城市</b></li>

         <li class="pure-menu-item"><a href="#" class="pure-menu-link">台北</a>
</li>
         <li class="pure-menu-item"><a href="#" class="pure-menu-link">新北</a>
</li>
         <li class="pure-menu-item"><a href="#" class="pure-menu-link">台中</a>
</li>
         <li class="pure-menu-item"><a href="#" class="pure-menu-link">台南</a>
</li>
         <li class="pure-menu-item"><a href="#" class="pure-menu-link">高雄</a>
</li>

         <li class="pure-menu-heading"><b>其他介绍</b></li>

         <li class="pure-menu-item"><a href="#" class="pure-menu-link">旅游休闲
</a></li>
         <li class="pure-menu-item"><a href="#" class="pure-menu-link">精致美食
</a></li>
         <li class="pure-menu-item"><a href="#" class="pure-menu-link">修身静心
</a></li>
       </ul>
    </div>
```

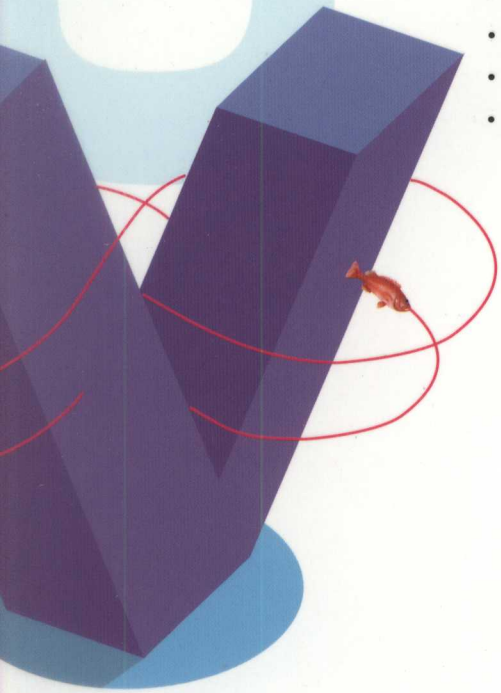
响应式网页设计实战

Responsive Web Design

响应式网页设计技术和方法会根据用户所使用设备的浏览器环境（例如屏幕的长度、宽度、长宽比、分辨率或设备屏幕显示的方向等）自动调整网页的版面，将恰当的内容和最佳的显示结果提供给用户。

本书内容重点

- 响应式网页设计概述
- 网页的新旧切版方式和字体资源
- 版面尺寸的固定方式与弹性方式
- Bootstrap 的学习与使用
- Foundation 的学习与使用
- Pure 的学习与使用



上架指导：网页设计

ISBN 978-7-111-54942-0

9 787111 549420 >

定价：49.00元

投稿热线：(010) 88379604
客服热线：(010) 88379426 88361066
购书热线：(010) 68326294 88379649 68995259

华章网站：www.hzbook.com
网上购书：www.china-pub.com
数字阅读：www.hzmedia.com.cn